UNE IMP EHE CWI ★QMIPS★ INR UER ZAR UTO

Quantitative
Modeling
In
Parallel
Systems

Part 3
Characteristic Equations for Markovian TIPP

Holger Hermanns Michael Rettelbach
University of Erlangen

Approved by: Ulrich Herzog

Abstract The stochastic process algebra MTIPP has emerged from research in the field of process descriptions for random behaviour through time. This calculus has recently been show to allow the calculation of performance measures (e.g. response times), purely functional statements (e.g. occurrences of deadlocks), as well as combined statements (e.g. optimal timeout values) [7, 8]. In contrast to classical process calculi each atomic action is supposed to happen after a delay that is characterised by a certain exponentially distributed random variable. In this report we present the language together with its operational semantics, that defines Markovian labelled transition systems as a combination of classical action-oriented transition systems and Markovian processes, especially continuous time Markov chains. In order to reflect different behavioural aspects we define a hierarchy of bisimulation equivalences and show that two of them are congruences. Finally we present characteristic equations for our central notion of equivalence, and show that these equations form a sound and complete axiomatisation for finite as well as infinite, regular processes. Theses equations characterise the behaviour of a MTIPP-processes on a syntactic level.

3.1 Introduction

Stochastic process algebras represent an approach to integrate qualitative analysis and performance evaluation into one comprehensive methodology [5, 9]. Classical process algebras are fairly well accepted means for the specification and qualitative analysis of distributed systems [10]. Performance evaluation on the other hand is usually carried out by monitoring, simulation and analytical modeling. While the former method supports testing and debugging a communication system, the latter two methods are suitable for performance prediction, tuning, and detection of performance bottlenecks. Various modeling methods suitable for analytical performance analysis are known, among them stochastic or timed extensions of finite state machines [2, 17, 15], stochastic Petri nets [1], or queueing networks [18].

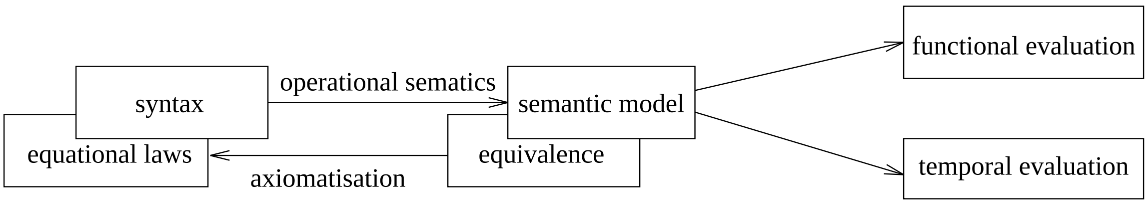

Figure 3.1: Overview of the stochastic process algebra approach

The main concept of the stochastic process algebra approach are shown in Figure 3.1. In this report we are concerned with the algebraic foundations of MTIPP[1]. Its *syntax* is given in terms of an abstract language, that is somehow between CCS and basic LOTOS, but in contrast to CCS and LOTOS our calculus is enriched with stochastic timing information. In order to model the system's development through time every action prefix is accompanied with a parameter, characterising its duration. Early work on TIPP has shown that general distributions are in principle possible [6], nevertheless current work is focussing on Markovian TIPP where these random variables are assumed to be exponentially distributed.

The *semantic model* is formally defined in a *structural operational* style introduced by Plotkin [16]. The fact that — like in every programming language — it is possible to write different programs that compute the same things, gives rise to a sophisticated *equivalence* notion. In our context this equivalence is *Markovian bisimulation* which characterises processes that behave in the same (stochastic) way. Additionally on the syntactic level we capture this behaviour by means of *equational laws* characterising Markovian bisimulation. *Functional* as well as *temporal evaluation* of an MTIPP description is usually carried out proceeding from the semantic model. But this subject is outside the scope of this paper, we refer the reader to [8].

The paper is organised as follows. Section 3.2 introduces basic definitions of the calculus together with its operational semantics. Section 3.3 establishes different notions of bisimulation suitable to characterise functional or temporal behaviour. Furthermore it is shown that *Markovian bisimulation* – our central notion of equivalence – is a congruence with respect to the language. Section 3.4 develops characteristic equations that form a sound and complete axiomatisation and elucidates the special algebraic properties of our calculus. Section 3.5 presents a brief summary and describes directions for further investigations.

[1] *M*arkovian *Ti*med *P*rocesses for *P*erformance Evaluation

$$\langle . \rangle \frac{}{(a,\lambda).P \xrightarrow{a,\lambda,\,\varepsilon} P}$$

$$\langle +_l \rangle \frac{P \xrightarrow{a,\lambda,\,w} P'}{P+Q \xrightarrow{a,\lambda,\,+_l.w} P'} \qquad \langle +_r \rangle \frac{Q \xrightarrow{a,\lambda,\,w} Q'}{P+Q \xrightarrow{a,\lambda,\,+_r.w} Q'}$$

$$\langle \|_l \rangle \frac{P \xrightarrow{a,\lambda,\,w} P'}{P\|_S Q \xrightarrow{a,\lambda,\,\|_l.w} P'\|_S Q} \quad (a \notin S) \qquad \langle \|_r \rangle \frac{Q \xrightarrow{a,\lambda,\,w} Q'}{P\|_S Q \xrightarrow{a,\lambda,\,\|_r.w} P\|_S Q'} \quad (a \notin S)$$

$$\langle \| \rangle \frac{P \xrightarrow{a,\lambda,\,v} P' \quad Q \xrightarrow{a,\mu,\,w} Q'}{P\|_S Q \xrightarrow{a,\lambda\mu,\,(v,w)} P'\|_S Q'} \quad (a \in S)$$

$$\langle rec \rangle \frac{P\{(recX : P)/X\} \xrightarrow{a,\lambda,\,w} P'}{recX \;:\; P \xrightarrow{a,\lambda,\,w} P'}$$

Figure 3.2: Operational semantics for MTIPP

3.2 The Language

3.2.1 Syntax: Markovian TIPP

We assume a fixed set of *action names* $Act := Com \cup \{\tau\}$, where we use τ as a distinguished symbol for internal, invisible activities and let Com be the set of regular, visible activities.

Definition 3.2.1 *$\mathcal{L}$ is defined as the set of closed terms that can be built by the following grammar:*

$$P \;::=\; 0 \;\mid\; (a,\lambda).P \;\mid\; P+P \;\mid\; P\|_S P \;\mid\; rec\ X : P \;\mid\; X \;,$$

where $a \in Act$, $\lambda \in I\!R^+$; $S \subseteq Act\backslash\{\tau\}$, $X \in Var$, and Var is a set of process variables.

The intuition behind the basic operators is refined with respect to TCSP: The *prefixed* process $(a,\lambda).P$ behaves as P after a duration that is exponentially distributed with parameter λ and has been accomplished by the instantaneous appearance of a. The process $P+Q$ (*Choice*) behaves as P or Q determined by the first action that appears. This implies that a minimum of several random time duration has to be calculated. Fortunately this is easy when dealing with exponential distributions; the according parameter is given by the sum of all involved parameters. This fact will be reflected in the axiomatization in Section 3.4.

Parallel composition of two processes is performed with the operator $\|_S$ where S is the set of actions to synchronize on. The parameter of a synchronized action is given by the product of the two parameters concerned. From a stochastic point of view this ensures that the resulting distribution remains proportional to its origins. The following section shows that the semantics of the other operators is defined as usual.

3.2.2 Semantics: Markovian Labelled Transition Systems

If an exponentially distributed duration is interrupted by the occurrence of an event, the remaining time is still exponentially distributed with exactly the *same* parameter. Thanks to this so-called *memoryless property* of the exponential distribution it is possible to model independent parallel execution of activities by interleaving. Thus the semantics of our language is given by a set of deduction rules :

Definition 3.2.2 *A* markovian labelled transition system (MLTS) *is a triple* $<\ \mathcal{L}\ ,\ P\ ,\ \longrightarrow\ >$ *where:*

- $\mathcal{L}$ *is the set of all* processes.
- P *is the* initial process.
- $\longrightarrow\ \subseteq (\,\mathcal{L} \times (Act \times I\!R^{+} \times Lab) \times \mathcal{L})$ *is the least relation*[2] *satisfying the rules of Figure 3.2.*

This MLTS embodies both, the temporal information represented by *positive reals* and the functional information by *actions*. The third label of each transition is a technical means to distinguish transitions that otherwise would be identified. The underlying idea is to explicitly produce different transitions if there are different deduction trees for a transition.

The process $(a, \lambda).0 + (a, \mu).0$, for example, is supposed to execute the action a after a delay that is distributed with the sum of λ and μ, as mentioned in Section 3.2.1. Semantically this is represented by two transitions, labeled with $(a, \lambda, +_l)$ and $(a, \mu, +_r)$, respectively. In order to achieve a similar representation for a process like $(a, \lambda).0 + (a, \lambda).0$, the third label becomes important to witness that there are two different deduction trees for (a, λ), yielding different transitions:

$$(a, \lambda).0 + (a, \lambda).0 \xrightarrow{a,\lambda,+_l} 0 \qquad \text{and} \qquad (a, \lambda).0 + (a, \lambda).0 \xrightarrow{a,\lambda,+_r} 0$$

This technique is strongly related to [3] and was already used in the probabilistic calculus PCCS [4] in a similar fashion.

3.3 Notions of Equivalence

Each MLTS embodies both, the functional as well as the temporal behaviour of a described system. Therefore notions of equivalence can either cover the functional matters, the temporal matters or even both of them.

3.3.1 Definitions

In the remainder of this paper we shall use the following notations:

Definition 3.3.1 *For $P, P' \in \mathcal{L}$, $a \in Act$, $\lambda \in I\!R^{+}$, $w \in Lab$ and $C \subseteq \mathcal{L}$ we define three abbreviations:*

1. $P \xrightarrow{a,\lambda} P' \quad :\Longleftrightarrow \quad \exists w \in Lab : P \xrightarrow{a,\lambda,w} P'$
2. $P \xrightarrow{a} P' \quad :\Longleftrightarrow \quad \exists \lambda \in I\!R^{+} : P \xrightarrow{a,\lambda} P'$
3. $P \xrightarrow{a,\lambda}\!\!\triangleright\, C \quad :\Longleftrightarrow \quad \exists P' \in C : P \xrightarrow{a,\lambda} P'$

From a purely functional point of view we can define classical strong bisimulation $\sim_F$ in the style of Park [14] using the abbreviations of Definition 3.3.1:

Definition 3.3.2 *A relation $\mathcal{B} \subseteq \mathcal{L} \times \mathcal{L}$ is a* **functional bisimulation** *if $P\,\mathcal{B}\,Q$ implies, that for all $a \in Act$*

(i) $\forall P' \in \mathcal{L} : \Big(P \xrightarrow{a} P' \quad \Longrightarrow \quad \exists Q' \in \mathcal{L} : Q \xrightarrow{a} Q' \wedge P'\,\mathcal{B}\,Q'\Big)$

(ii) $\forall Q' \in \mathcal{L} : \Big(Q \xrightarrow{a} Q' \quad \Longrightarrow \quad \exists P' \in \mathcal{L} : P \xrightarrow{a} P' \wedge Q'\,\mathcal{B}\,P'\Big)$

Two processes P and Q are functional bisimulation equivalent ($P \sim_F Q$) if there exists a functional bisimulation $\mathcal{B}$ such that $P\,\mathcal{B}\,Q$.

[2] *Lab* is the set of words defined by the following grammar: $w := \varepsilon \mid +_l .w \mid +_r .w \mid \|_l.w \mid \|_r.w \mid (w, w)$.

In order to take into consideration both, the functional and the temporal behaviour, we have to define a combined bisimulation. It is no longer sufficient that both processes can perform the same actions to change into equivalent states. Moreover the rates to change into those states have to be considered.

Regard for example the process $(a, \lambda).(P + Q) + (a, \mu).(Q + P)$. From a stochastic point of view the behaviour of this process is considered to be identical to that of $(a, \lambda + \mu).(P + Q)$. In general, the *sum* of parameters to *equivalent* states is the decisive value. As we are interested in a relation that also covers the functional aspects of behavior this has to be considered for each distinct action.

This aim is accomplished by an equivalence based on Larsen and Skou's *probabilistic bisimulation* [11] that we call *markovian* with respect to the context of this paper[3].

Definition 3.3.3 *The function* $\gamma : (\mathcal{L} \times Act \times 2^{\mathcal{L}}) \longrightarrow I\!R^+$ *is defined as follows:*

$$\gamma(P, a, C) := \sum \{\!|\, \lambda \mid P \xrightarrow{a,\lambda} C \,|\!\} \tag{3.1}$$

Definition 3.3.4 *An equivalence relation* $\mathcal{B} \subseteq \mathcal{L} \times \mathcal{L}$ *is a* **markovian bisimulation** *if* $P \mathcal{B} Q$ *implies*

$$\forall C \in \mathcal{L}/_{\mathcal{B}} : \forall a \in Act : \gamma(P, a, C) = \gamma(Q, a, C) \tag{3.2}$$

Two processes P *and* Q *are markovian bisimulation equivalent (written* $P \sim_M Q$*) if there exists a markovian bisimulation* $\mathcal{B}$ *such that* $P \mathcal{B} Q$.

Moreover an observer might only be interested in the temporal behavior of a system, i.e., disregarding the action names, just concerning (random) durations between certain activities. Exactly those aspects are described by Markov chains. In order to reflect them in our calculus, we define *temporal bisimulation* ($\sim_T$) simply by coarsening markovian bisimulation:

Definition 3.3.5 *A relation* $\mathcal{B} \subseteq \mathcal{L} \times \mathcal{L}$ *is a* **temporal bisimulation** *if* $P \mathcal{B} Q$ *implies*

$$\forall C \in \mathcal{L}/_{\mathcal{B}} : \quad \sum_{a \in Act} \gamma(P, a, C) = \sum_{a \in Act} \gamma(Q, a, C) \tag{3.3}$$

Two processes P *and* Q *are temporal bisimulation equivalent (written* $P \sim_T Q$*) if there exists a temporal bisimulation* $\mathcal{B}$ *such that* $P \mathcal{B} Q$.

The relationship between these definitions is subsumed by the following proposition: Markovian bisimulation equivalence is a strict refinement of temporal and functional bisimulation equivalence:

Proposition 3.3.1 $\quad \sim_M \subset (\sim_F \cap \sim_T)$

Proof: Let $P \sim_M Q$, i.e. $\gamma(P, a, C) = \gamma(Q, a, C)$ for any action a and equivalence class C. Hence it follows that $\sum_{a \in Act} \gamma(P, a, C) = \sum_{a \in Act} \gamma(Q, a, C)$, leading to the observation that $\sim_M$ itself is a *temporal* bisimulation, whence $P \sim_T Q$ follows immediately.

In order to deduce $P \sim_F Q$, we restrict ourselves to show part (i) of Definition 3.3.2: Given that $P \xrightarrow{a} P'$, we know, by assumption, that $0 < \gamma(P, a, [P']_{\sim_M}) = \gamma(Q, a, [P']_{\sim_M})$. Thus there exists a process $Q' \in [P']_{\sim_M}$ such that $Q \xrightarrow{a} Q'$.

In order to show that $\sim_F$ and $\sim_T$ are *strictly* coarser than $\sim_M$ we consider $P := (a, \lambda).0 + (b, 2\lambda).0$ and $Q := (a, 2\lambda).0 + (b, \lambda).0$. It is easy to check that $P \sim_F Q$ and $P \sim_T Q$, but $P \not\sim_M Q$, completing the proof.

□

[3] In the following definitions we use $\mathcal{L}/_{\mathcal{B}}$ to denote the set of equivalence classes induced by an equivalence relation $\mathcal{B}$ over $\mathcal{L}$. Moreover, for any process P the according equivalence class is denoted by $[P]_{\mathcal{B}}$. The symbols $\{\!|$ and $|\!\}$ are used as multiset brackets.

3.3.2 Congruences

Like strong bisimulation does for CCS, the main equivalences defined above provide a compositional notion of semantics for MTIPP that is consistent with the operational semantics defined in the last section. Specifically two of them are congruences. For functional bisimulation this is not surprising as there is no significant difference to standard strong bisimulation for CCS. We will show this congruence result for markovian bisimulation, temporal bisimulation is not a congruence, as synchronization takes place by considering action names. A simple counterexample is: $(a,\lambda).0 \sim_T (b,\lambda).0 \quad \not\Rightarrow$ $(a,\lambda).0 \;\|_{\{a,b\}}(a,\mu).0 \quad \sim_T \quad (b,\lambda).0 \;\|_{\{a,b\}}(a,\mu).0$

Proposition 3.3.2 $\quad \sim_M$ *is a congruence with respect to all operators in* $\mathcal{L}$.

Proof: We restrict ourselves to show the case of parallel composition. The proofs for the other operators are similar.

Assuming that $P \sim_M Q$, we show that $P\|_S R \sim_M Q\|_S R$. It is enough to show that the reflexive closure $\mathcal{B}$ of

$$\mathcal{R} := \{(P\|_S R, Q\|_S R) \mid P \sim_M Q \;\wedge\; R \in \mathcal{L}\}$$

is a markovian bisimulation, i.e. that for arbitrary $a \in Act$ and $C \in \mathcal{L}/_{\mathcal{B}}$ it follows from $P \sim_M Q$ that

$$\gamma(P\|_S R, a, C) = \gamma(Q\|_S R, a, C) \tag{3.4}$$

We distinguish the cases whether a is an action to synchronize on or not. Furthermore we assume that C includes an element $\widetilde{P}\|_S\widetilde{R}$, because otherwise C contains exactly one element, that can't be a successor state of $P\|_S R$.

$a \notin S$: By Definition 3.3.3 and 3.3.1 as well as the operational rules $\langle\|_r\rangle$ and $\langle\|_l\rangle$ we obtain

$$\begin{aligned}\gamma(P\|_S R, a, [\widetilde{P}\|_S\widetilde{R}]_{\mathcal{B}}) \;&=\; \sum\{\!|\lambda \mid \exists\, P'\|_S R \in [\widetilde{P}\|_S\widetilde{R}]_{\mathcal{B}} \;:\; P\|_S R \xrightarrow{a,\lambda} P'\|_S R|\!\} \\ &\quad +\sum\{\!|\lambda \mid \exists\, P\|_S R' \in [\widetilde{P}\|_S\widetilde{R}]_{\mathcal{B}} \;:\; P\|_S R \xrightarrow{a,\lambda} P\|_S R'|\!\}\end{aligned}$$

Observing that $P'\|_S R \in [\widetilde{P}\|_S R]_{\mathcal{B}} \iff P' \in [\widetilde{P}]_{\sim_M}$ the sum above coincides with

$$\sum\{\!|\lambda \mid \exists\, P' \in [\widetilde{P}]_{\sim_M} : P \xrightarrow{a,\lambda} P'|\!\} + \sum\{\!|\lambda \mid \exists\, R' \in [\widetilde{R}]_{\sim_M} : R \xrightarrow{a,\lambda} R'|\!\},$$

whence, by the reverse steps it follows that $\gamma(P\|_S R, a, [\widetilde{P}\|_S\widetilde{R}]_{\mathcal{B}}) = \gamma(P, a, [\widetilde{P}]_{\sim_M}) + \gamma(R, a, [\widetilde{R}]_{\sim_M})$. By symmetric arguments we obtain $\gamma(Q\|_S R, a, [\widetilde{P}\|_S\widetilde{R}]_{\sim_M}) = \gamma(Q, a, [\widetilde{P}]_{\sim_M}) + \gamma(R, a, [\widetilde{R}]_{\sim_M})$. Using the assumption $P \sim_M Q$, especially $\gamma(P, a, [\widetilde{P}]_{\sim_M}) = \gamma(Q, a, [\widetilde{P}]_{\sim_M})$ concludes the proof in this case.

$a \in S$: Similar to the above we use Definition 3.3.3, 3.3.1 and rule $\langle\|\rangle$ in order to deduce $\gamma(P\|_S R, a, [\widetilde{P}\|_S\widetilde{R}]_{\mathcal{B}}) = \gamma(P, a, [\widetilde{P}]_{\sim_M})\;\gamma(R, a, [\widetilde{R}]_{\sim_M})$ as well as $\gamma(Q\|_S R, a, [\widetilde{P}\|_S\widetilde{R}]_{\sim_M}) = \gamma(Q, a, [\widetilde{P}]_{\sim_M})\;\gamma(R, a, [\widetilde{R}]_{\sim_M})$. Again, by using $P \sim_M Q$, we convince ourselves that both products are identical.

□

3.4 Characteristic Equations

As markovian bisimulation equivalence ($\sim_M$) is our central notion of equivalence we develop a set of equational laws for the calculus MPA which are sound and complete with respect to $\sim_M$. We shall use $\equiv$ to represent syntactic identity, and $=$ to represent derivability in our equational theory.

3.4.1 Sequential Processes

Let us start with a subset $\mathcal{L}_{seq}$ of MTIPP that is defined by the following grammar:

$$P \; := \; 0 \quad | \quad (a, \lambda).P \quad | \quad P + P$$

The laws presented in Figure 3.3 form a sound and complete axiomatization of this (core) language.

$$\begin{array}{rcll}
P + 0 & = & P & \langle +_0 \rangle \\
P + Q & = & Q + P & \langle +_K \rangle \\
(P + Q) + R & = & P + (Q + R) & \langle +_A \rangle \\
(a\,,\,\lambda).P + (a\,,\,\mu).P & = & (a\,,\,\lambda + \mu).P & \langle \lambda\mu \rangle
\end{array}$$

Figure 3.3: Axiomatization for $\mathcal{L}_{seq}$

The first three laws ($\langle +_0 \rangle$, $\langle +_K \rangle$, and $\langle +_A \rangle$) are well known from classical calculi. The $\langle \lambda\mu \rangle$ law, however, is a special feature of our calculus and can be regarded as a stochastic replacement of the conventional law for the idempotence of a choice operator ($P + P = P$). It reflects the nondeterminism of choice in our calculus which is modelled by the minimum of two random distributions. As already mentioned, this leads to an exponential distribution parameterized by the sum of the initial parameters.

If a MLTS contains two parallel[4] transitions labelled with the same action a, they can be substituted by a single one. According to this law, the appropriate parameter is given by the sum of the parameters of the replaced transitions. By iterative applications of this law, multiple occurences of parallel transitions can be subsumed.

As our MLTS keeps more information than a Markov chain, especially action names associated to single transitions, it distinguishes more transitions. Therefore we are only allowed to treat transitions labelled with the same action names in that way (see also Figure 3.4).

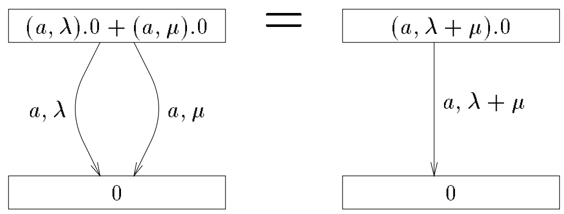

Figure 3.4: An example for the $\langle \lambda\mu \rangle$ law

Indeed, if we have shown the completeness of axiomatization, the $\langle \lambda\mu \rangle$ law serves to accumulate *all* transitions with a special action a from one state to *all states* that belong to the same equivalence class. Consider, for example, a process like $(a\,,\,\lambda).P + (a\,,\,\mu).Q$, where P and Q are equivalent. Thanks to the completeness Q can surely be transformed into P. Afterwards the $\langle \lambda\mu \rangle$ law is applicable, leading to $(a\,,\,\lambda + \mu).P$.

Soundness

Proposition 3.4.1 *The laws presented in Figure 3.3 form a sound axiomatization of* $\sim_M$ *with respect to all operators of* $\mathcal{L}$.

[4]We refer to transitions with the same source and destination states as *parallel* transitions.

Proof: The proof is straightforward. As an example we present some details concerning $\langle\lambda\mu\rangle$:

Let $a \in Act$, $\lambda, \mu \in \mathbb{R}^+$ and $P \in \mathcal{L}$. For an arbitrary action b and equivalence class C it has to be shown that $\gamma((a,\lambda).P + (a,\mu).P, b, C) = \gamma((a,\lambda+\mu).P, b, C)$. If $b \neq a$ or $P \notin C$ then, obviously, $\gamma((a,\lambda).P + (b,\mu).P, b, C) = 0 = \gamma((a,\lambda+\mu).P, b, C)$. Otherwise it is easy to derive that $\gamma((a,\lambda).P + (a,\mu).P, a, [P]_\sim) = \lambda + \mu = \gamma((a,\lambda+\mu).P, a, [P]_\sim)$.

□

Completeness

The more gripping result is the completeness of our axiomatization. We define a normal form (NF) which is to some extent simpler than an arbitrary term, i.e. it is minimal with respect to the laws $\langle\lambda\mu\rangle$ and $\langle +_0\rangle$. Afterwards it is shown, that each process in $\mathcal{L}_{seq}$ can be transformed into a NF by application of the four laws.

Finally we show completeness for processes in NF. Because of the soundness we obtain the completeness result by using transitivity of the relations involved. A similar proof technique can be found e.g. in [13].

Definition 3.4.1 *The relation $\widehat{=} \subseteq \mathcal{L}_{seq} \times \mathcal{L}_{seq}$ is defined as follows:*
$P \widehat{=} Q \;\; :\Longleftrightarrow$ *P can be transformed into Q only by applying the laws $\langle +_K\rangle$ and $\langle +_A\rangle$.*

Lemma 3.4.1 *$\widehat{=}$ is an equivalence.*

Definition 3.4.2 *A process P is said to be in normal form, iff*

$$P \equiv 0 \qquad or \qquad P \equiv \sum (a_i\,,\,\lambda_i).P_i$$

where P_i is in NF and for $i \neq j$: $a_i = a_j \Rightarrow P_i \not\widehat{=} P_j$.

The intuition of a normal form process term is to represent the *minimal* (concerning the number of transitions) transition system within one equivalence class. In the remainder of this section we will use an underscore to explicitly denote that a process $\underline{P}$ is in NF.

Proposition 3.4.2 *For each $P \in \mathcal{L}$ there exists a NF term $\underline{P} \in \mathcal{L}$ with $\underline{P} = P$.*

Proof: By structural induction on P.

$P \equiv 0$: P is in NF.

$P \equiv (a,\lambda).P'$: By induction, there is a $\underline{P'} = P'$ in NF, so $\underline{P} \equiv (a,\lambda).\underline{P'}$ is in NF and $P = \underline{P}$.

$P \equiv Q + R$: By induction there are $\underline{Q}$ and $\underline{R}$ in NF with $Q = \underline{Q}$ and $R = \underline{R}$. Let us now assume, that one of both $\underline{Q}$ and $\underline{R}$ is the *stop* process 0, say $\underline{R}$. Then we have $P = \underline{Q}$, because of the law $\langle +_0\rangle$.

Otherwise, if both are different from 0, they can be written as

$$\underline{Q} \equiv \sum (a_i\,,\,\lambda_i).\underline{Q_i} \qquad \underline{R} \equiv \sum (b_i\,,\,\mu_i).\underline{R_i}$$

Searching for a NF-term for P the following term $\widetilde{P}$ seems to be an acceptable candidate:

$$\widetilde{P} \equiv \sum (a_i\,,\,\lambda_i).\underline{Q_i} + \sum (b_i\,,\,\mu_i).\underline{R_i}$$

But the side conditon of Definition 3.4.2 may be violated, because there might be actions $a_i = b_j$ where $\underline{Q_i} \widehat{=} \underline{R_j}$ (cf. Figure 3.5 for an example). Transforming Q_i into R_j and applying the $\langle\lambda\mu\rangle$ law we can eliminate this violation of the side condition. By repeating this transformation we obtain a NF term (cf. Figure 3.6) after a finite number of steps, because $\widetilde{P}$ is finite.

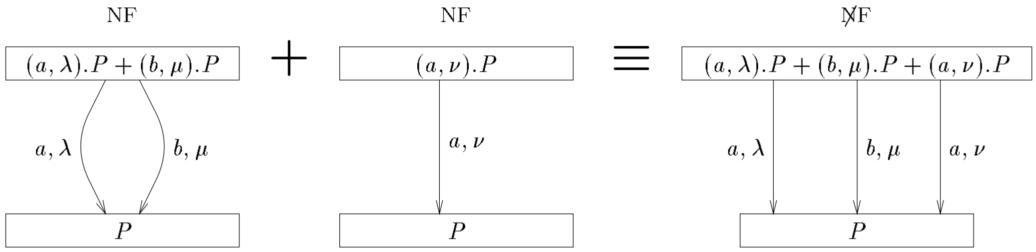

Figure 3.5: Violation of the side condition

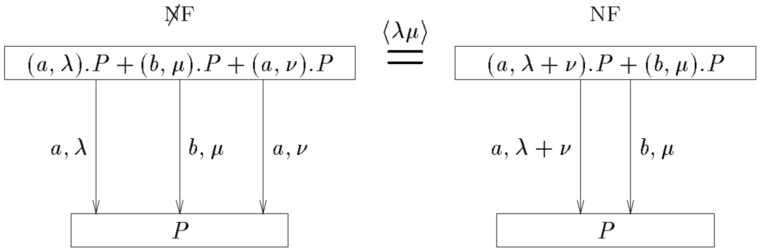

Figure 3.6: Assuring the side condition

□

We proceed by demonstrating that two equivalent normal forms $\underline{P}$ and $\underline{Q}$ can be transformed into each other only by application of $\langle +_A \rangle$ and $\langle +_K \rangle$. Actually this implies $\underline{P} = \underline{Q}$.

Lemma 3.4.2 *Let $\underline{P} \equiv \sum (a_i, \lambda_i).\underline{P_i}$ be a NF term, so the following proposition holds:*

$$\forall b \in Act : \forall S \in \mathcal{L}/\!\sim_M : \qquad \Big(\exists k : (a_k = b \wedge \underline{P_k} \in S) \iff \gamma(\underline{P}, b, S) > 0 \Big)$$

Proof: The lemma follows immediately from Definition 3.3.3 and the deduction rules of Figure 3.2.
□

Proposition 3.4.3 $\underline{P} \sim_M \underline{Q} \implies \underline{P} \hat{=} \underline{Q}$

Proof: By induction on the sum of sizes of $\underline{P}$ and $\underline{Q}$. Assuming $\underline{P} \sim_M \underline{Q}$ we know for arbitrary action b and equivalence class $S \in \mathcal{L}/\!\sim_M$ that $\gamma(\underline{P}, b, S) = \gamma(\underline{Q}, b, S)$ holds.

$\underline{P} \equiv 0$: This implies $\gamma(\underline{P}, b, S) = 0$ for any b and S. By assumption it follows that $\gamma(\underline{Q}, b, S) = 0$, thus giving $\underline{Q} \equiv 0$.

$\underline{P} \equiv \sum (a_i, \lambda_i).\underline{P_i}$: From Lemma 3.4.2 it follows that for any index i there is a positive value $\lambda^* > 0$ such that $\lambda^* = \gamma(\underline{P}, a_i, [\underline{P_i}]_\sim)$. Using Definition 3.3.3 and 3.3.1 it follows that $\lambda^* = \sum \{\!| \lambda \mid \exists j : \underline{P_j} \sim_M \underline{P_i} \wedge \underline{P} \xrightarrow{a_i, \lambda} \underline{P_j} |\!\}$. But then, by induction, $\lambda^* = \sum \{\!| \lambda \mid \exists j : \underline{P_j} \hat{=} \underline{P_i} \wedge \underline{P} \xrightarrow{a_i, \lambda} \underline{P_j} |\!\}$. Because of the side condition of Definition 3.4.1 there is exactly *one* such j, namely i. Thus the multiset $\{\!| \lambda \mid \exists j : \underline{P_j} \hat{=} \underline{P_i} \wedge \underline{P} \xrightarrow{a_i, \lambda} \underline{P_j} |\!\}$ consists of *exactly one* element, that is λ_i. In short:

$$\forall i : \gamma(\underline{P}, a_i, [\underline{P_i}]_\sim) = \lambda_i$$

By reasoning similar as above we obtain:

$$\forall j : \gamma(\underline{Q}, b_j, [\underline{Q_j}]_\sim) = \mu_j$$

Due to our assumption $\underline{P} \sim_M \underline{Q}$ we conclude:

$$\forall i : \exists j : \lambda_i = \mu_j \wedge a_i = b_j \wedge \underline{P}_i \sim_M \underline{Q}_j$$
$$\forall j : \exists i : \lambda_i = \mu_j \wedge a_i = b_j \wedge \underline{P}_i \sim_M \underline{Q}_j$$

By induction we finally obtain, that $\underline{P}$ and $\underline{Q}$ differ from each other only with respect to the ordering of their operands, establishing our proposition $\underline{P} \widehat{=} \underline{Q}$:

$$\forall i : \exists j : \lambda_i = \mu_j \wedge a_i = b_j \wedge \underline{P}_i \widehat{=} \underline{Q}_j$$
$$\forall j : \exists i : \lambda_i = \mu_j \wedge a_i = b_j \wedge \underline{P}_i \widehat{=} \underline{Q}_j$$

□

Corollary 3.4.1 *The laws presented in Figure 3.3 form a complete axiomatization of $\sim_M$ for $\mathcal{L}_{seq}$, i.e. $P \sim_M Q$ implies $P = Q$.*

Proof: Assume $P \sim_M Q$. By Proposition 3.4.2, we can find $\underline{P}$ and $\underline{Q}$, both in NF form such that $P = \underline{P}$ and $Q = \underline{Q}$. Using Proposition 3.4.1 as well as the transitivity of $\sim_M$ we obtain $\underline{P} \sim_M \underline{Q}$. But then by Proposition 3.4.3, we must have $\underline{P} = \underline{Q}$, whence we conclude $P = Q$. □

3.4.2 Regular Processes

In this section, we shall add recursion to the language, yielding a class of processes usually known as regular processes. We will show that the laws of Figure 3.7 together with those of Figure 3.3 form a sound and complete theory for this extended language.

$rec\ X : P$	$=$	$rec\ Y : P\{Y/X\}$	Y not free in $rec\ X : P$	$\langle R1 \rangle$
$rec\ X : P$	$=$	$P\{rec\ X : P/X\}$		$\langle R2 \rangle$
$rec\ X : P$	$=$	$rec\ X : (P + X)$		$\langle R3 \rangle$
if $Q = P\{Q/X\}$	then	$Q = rec\ X : P$	X guarded in P	$\langle R4 \rangle$

Figure 3.7: Axioms for recursion

It is easy to show that these laws respect markovian bisimulation equivalence, i.e. they are sound. Here we shall present the proof of completeness for regular processes. The structure of the proof closely follows the lines of Milner's for CCS [12] except for some additional arguments necessary because of the rates of each action.

The structure of the proof is as follows: First of all we shall show that every (regular) process P provably satisfies a certain set of mutual recursive equations. Next we will see that if $P \sim_M Q$ then there is a *common* equation set provably satisfied by each of P and Q. In fact, this is the crucial part of the proof, in contrast to the last step. There we show that if P and Q provably satisfy the same equation set, then it follows that $P = Q$.

Definition 3.4.3 *An* equation set $\vec{X} = \vec{P}$ *is a finite non empty sequence of declarations $X_1 = P_1, \ldots, X_n = P_n$, where the X_is are pairwise distinct process variables and P_is are regular processes.*

A vector $\vec{Q}$ satisfies $\vec{X} = \vec{P}$ iff $\forall i : Q_i \sim_M P_i\{\vec{Q}/\vec{X}\}$.

A vector $\vec{Q}$ provably satisfies $\vec{X} = \vec{P}$ iff $\forall i : Q_i = P_i\{\vec{Q}/\vec{X}\}$.

A process Q provably satisfies $\vec{X} = \vec{P}$ iff we can find a vector $\vec{Q}$ which provably satisfies $\vec{X} = \vec{P}$ and $Q = Q_1$.

Definition 3.4.4 *An equation set $\vec{X} = \vec{P}$ is standard iff each P_i is of the form:*

$$\sum_{j=1}^{n(i)}(a_j, \lambda_j).V_{f(i,j)} + \sum_{k=1}^{m(i)} W_{g(i,k)}$$

where the vectors $\vec{V}$ and $\vec{W}$ are disjoint. We shall call $\vec{V}$ the formal *variables and $\vec{W}$ the* free *variables of $\vec{X} = \vec{P}$.*

Proposition 3.4.4 *For any P we can find a standard equation set $\vec{X} = \vec{Q}$ which P provably satisfies.*

Proof: See Milner [12, Theorem 5.9]. This is Milner's Equational Characterization Theorem. The proof essentially shows that the law $\langle R4 \rangle$ is also applicable to sets of equations. The fact that markovian bisimulation is a refinement of strong bisimulation does not affect the proof. □

Proposition 3.4.5 *If $P \sim_M Q$ then there is a standard equation set $\vec{Y} = \vec{R}$ provably satisfied by both, P and Q.*

Proof: The proof is a slight modification of Milner's [12, Theorem 5.10]. By Proposition 3.4.4 there are standard equation sets $\vec{X}^P = \vec{P}$ and $\vec{X}^Q = \vec{Q}$ provably satisfied by P and Q respectively. To enhance clarity we assume that P and Q are closed, i.e. they do not contain free variables. The opposite case is widely discussed in [12]. Thus, by Definition 3.4.3 and 3.4.4, we have $P = X_1^P$ and $Q = X_1^Q$, and moreover:

$$X_{i^P}^P \equiv \sum_{j}^{n(i^P)} (a_j, \lambda_j).X_{f(i^P,j)}^P \tag{3.5}$$

$$X_{i^Q}^Q \equiv \sum_{k}^{n(i^Q)} (b_k, \mu_k).X_{f(i^Q,k)}^Q \tag{3.6}$$

We now consider the relation $I = \{(i^P, i^Q) \mid X_{i^P}^P \sim_M X_{i^Q}^Q\}$. Because of the soundness of our axioms, $(1,1)$ is surely an element of I. However also other pairs of indices must be in I: Whenever (i^P, i^Q) is in I, the indices of equivalent subprocesses of $X_{i^P}^P$ and $X_{i^Q}^Q$ must also be in I. We formalize this observation by a set of relations $J_{i^P i^Q} \subseteq \{1, \ldots n(i^P)\} \times \{1, \ldots n(i^Q)\}$ defined for every pair (i^P, i^Q) as follows:

$$J_{i^P i^Q} = \{(j,k) \mid a_j = b_k \ \wedge \ (f(i^P, j), f(i^Q, k)) \in I\}$$

Since $X_{i^P}^P \sim_M X_{i^Q}^Q$ we can be sure that every subprocess of $X_{i^P}^P$ and $X_{i^Q}^Q$ is included somewhere in $J_{i^P i^Q}$, i.e. $J_{i^P i^Q}$ is total and surjective. Furthermore it is not hard to see that the sum of rates to equivalent subprocesses can easily computed by means of the relation $J_{i^P i^Q}$:

$$\forall j \in \{1, \ldots n(i^P)\} \ : \ \gamma(X_{i^P}^P, a_j, [X_{f(i^P,j)}]\sim) = \sum_{(j,k) \in J_{i^P i^Q}} \mu_k \tag{3.7}$$

$$\forall k \in \{1, \ldots n(i^Q)\} \ : \ \gamma(X_{i^Q}^Q, b_k, [X_{f(i^Q,j)}]\sim) = \sum_{(j,k) \in J_{i^P i^Q}} \lambda_j \tag{3.8}$$

Remark that, by Definition 3.3.4, $X_{i^P}^P \sim_M X_{i^Q}^Q$ implies that the left-hand sides of the two equations above have to be identical. Thus the sums on the right-hand are identical to. In the following we denote this sum γ_{jk}.

After these introductorial definitions we are now able to define the common equation set that is central for the proof: Assume for every $(i^P, i^Q) \in I$ a new variable $Y_{i^P i^Q}$. Define for each $(i^P, i^Q) \in I$ the formal equations $Y_{i^P i^Q} = R_{i^P i^Q}$ as follows:

$$Y_{i^P i^Q} = \sum_{(j,k) \in J_{i^P i^Q}} \left(a_{jk}, \frac{\lambda_j \mu_k}{\gamma_{jk}} \right) .Y_{f(i^P,j) f(i^Q,k)} \tag{3.9}$$

This equation set $Y = \vec{R}$ is in standard form, furthermore we can show that it provably satisfies the vector $\vec{X}^P$. By Definition 3.4.3 this is true iff for all i^P and i^Q

$$X_{i^P} = R_{i^P i^Q}\{\vec{X}/\vec{Y}\} \tag{3.10}$$

holds. To see this, we observe that

$$R_{i^P i^Q}\{\vec{X}/\vec{Y}\} \equiv \sum_{(j,k) \in J_{i^P i^Q}} \left(a_{jk}, \frac{\lambda_j \mu_k}{\gamma_{jk}} \right) .X_{f(i^P,j)} \tag{3.11}$$

includes all subprocesses $X_{f(i^P,j)}$ of X_{i^P}, since $J_{i^P i^Q}$ is total, but compared with 3.5 some of them may be repeated. Additionally the rates λ_j are split up on these repeated processes wheighted by the factors $\frac{\mu_k}{\gamma_{jk}}$. Nevertheless these differences vanish by — probably multiple — applications of the law $\langle \lambda\mu \rangle$ to amalgamate repeated processes. Thus it is possible to prove that $X_{i^P} = R_{i^P i^Q}\{\vec{X}/\vec{Y}\}$.

By symmetric arguments, since $J_{i^P i^Q}$ is surjective, we can argue that the vector $\vec{X}^Q$ provably satisfies $Y = \vec{R}$ as well. □

Proposition 3.4.6 *If P and Q provably satisfy the same standard equation set then $P = Q$.*

Proof: Let $\vec{Y} = \vec{R}$ be the standard equation set provably satisfied by P and Q. Definition 3.4.3 assures that $P = R_1$ and $Q = R_1$. The result follows immediate. □

Corollary 3.4.2 *The laws presented in Figure 3.3 form a complete axiomatization of $\sim_M$ for regular MTIPPprocesses, i.e. $P \sim_M Q$ implies $P = Q$.*

Proof: By Proposition 3.4.5 there is is a standard equation set $\vec{Y} = \vec{R}$ provably satisfied by both, P and Q. By Proposition 3.4.6 it follows that P = Q. □

3.5 Conclusion and Prospects

In this paper we have proposed MTIPP, a calculus to describe markovian processes. Under different behavioural viewpoints we have systematically studied the issues of bisimulation equivalence, congruence and complete axiomatization. Arising from markovian labeled transition systems we defined markovian bisimulation as our central equivalence notion. In particular we included a sound and complete axiomatization of markovian bisimulation equivalence for finite as well as infinite,regular processes.

Currently we are developing extensions to more general distributions in order to enhance the expressiveness of the calculus as well as performance measurement formalisms working on MLTS.

As our calculus embodies an implicit notion of elapsing time, it is interesting to investigate the relation to timed calculi like Wangs CCS derivative [19]. Especially dealing with deterministic durations instead of exponential distributed ones might partially bridge the gap. This issue remains open.

Acknowledgements

Special thanks to N. Götz, U. Herzog, and V. Mertsiotakis for proof reading and helpful comments.

Bibliography

[1] M. Ajmone Marsan, G. Balbo, and G. Conte. A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems. *ACM Transactions on Computer Systems*, 2(2):93–122, May 1984.

[2] F. Bause and P. Buchholz. Protocol Analysis using a timed version of SDL. In *Proc. of 3rd. International Conference on Formal Description Techniques (FORTE '90)*, Madrid, Spain, November 1990.

[3] G. Boudol and I. Castellani. A non-interleaving semantics for ccs based on proved transitions. *Fundamenta Informaticae*, XI:433–452, 1988.

[4] A. Giacalone, C.-C. Jou, and S.A. Smolka. Algebraic reasoning for probabilistic concurrent systems. In *Proceedings of Working Conference on Programming Concepts and Methods*, Sea of Gallilee, Israel, April 1990. IFIP TC 2.

[5] 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 *Proc. of the 16th Int'l Symposium on Computer Performance Modelling, Measurement and Evaluation, PERFORMANCE '93*. Springer, 1993. LNCS 729.

[6] Norbert Götz, Ulrich Herzog, and Michael Rettelbach. TIPP — introduction and application to protocol performance analysis. In *Formale Beschreibungstechniken für verteilte Systeme*, Munich, to appear 1993. FOKUS series, Saur publishers.

[7] H. Hermanns, U. Herzog, J. Hillston, V. Mertsiotakis, and M. Rettelbach. Stochastic Process Algebras: Integrating Qualitative and Quantitative Modelling. Technical Report 11/94, Universität Erlangen–Nürnberg, IMMD VII, Martensstr. 3, 91058 Erlangen, May 1994.

[8] U. Herzog and V. Mertsiotakis. Applying Stochastic Process Algebras to Failure Modelling. In *Proc. of the 2nd Workshop on Process Algebras and Performance Modelling*, Regensberg, Germany, July 1994.

[9] J. Hillston. *A Compositional Approach to Performance Modelling.* PhD thesis, University of Edinburgh, 1994.

[10] I.S.O. *LOTOS : A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour.* ISO, 1989.

[11] K.G. Larsen and A. Skou. Bisimulation through probabilistic testing. In *Proc. of 16th ACM Symposium on Principles of Programming Languages*, pages 344–352, 1989.

[12] Robin Milner. A complete inference system for a class of regular behaviours. *J. Comput. Sytem Sci.*, 28:439–466, 1984.

[13] Faron Moller and Chris Tofts. A Temporal Calculus of Communicating Systems. Technical Report ECS-LFCS-89-104, University of Edinburgh, December 1989.

[14] D. Park. Concurrency and automata on infinite sequences. In *Proc. of 5th GI Conference on Theoretical Computer Science*, pages 167–183. Springer, 1993. LNCS 104.

[15] B. Plateau, J-M. Fourneau, and K-H. Lee. PEPS: A Package for Solving Complex Markov Models of Parallel Systems. In R. Puigjaner, editor, *Proceedings of the 4th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, pages 341–360. Planum Press, 1988.

[16] Gordon Plotkin. A structural approach to operational semantics. Report DAIMI FN-19, Computer Science Department, Aarhus University, September 1981.

[17] H. Rudin. From Formal Protocol Specification Towards Automated Performance Prediction. In H. Rudin and C.H. West, editors, *Protocol Specification, Testing and Verification*, volume III, pages 257–269. North Holland (IFIP), 1983.

[18] M. Sczittnick and B. Müller-Clostermann. MACOM - A Tool for the Markovian Analysis of Communication Systems. In *Proceedings of the 4th International Conference on Data Communication Systems and their Performance*, Barcelona, Spain, June 1990.

[19] Wang Yi. CCS + Time = an Interleaving Model for Real Time Systems. In , editor, *Proceedings of ICALP '91*, pages 217–228, 1991.