ASCENS
Autonomic Service-Component Ensembles

TR 09: Coalgebraic Bisimulation of FuTS

Grant agreement number: 257414
Funding Scheme: FET Proactive
Project Type: Integrated Project
Latest version of Annex I: 7.6.2010

Author(s): D. Latella (ISTI), M. Massink (ISTI), E. P. de Vink (Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven and Centrum Wiskunde en Informatica, Amsterdam)

Date of technical report: January 16, 2013
Revision: V1
Classification: PU

Project coordinator: Martin Wirsing (LMU)
Tel: +49 89 2180 9154
Fax: +49 89 2180 9175
E-mail: wirsing@lmu.de

Partners: LMU, UNIPI, UDF, Fraunhofer, UJF-Verimag, UNIMORE, ULB, EPFL, VW, Zimory, UL, IMT, Mobsya, CUNI
Coalgebraic Bisimulation of FuTS

January 16, 2013

abstract

Labeled state-to-function transition systems, FuTSs for short, capture transition schemes incorporating multiplicities from states to functions of finite support over general semirings. As such FuTSs constitute a convenient modeling instrument to deal with process languages and their stochastic extensions in particular. In this paper, the notion of bisimulation induced by a FuTS is addressed from a coalgebraic point of view. A correspondence result is established stating that FuTS-bisimilarity coincides with behavioural equivalence of the associated functor. Moreover, it is shown that for FuTSs involving a specific type of semiring only, weak pullbacks are preserved. As a consequence, for these FuTSs, behavioural equivalence coincides with coalgebraic bisimilarity. As generic examples, the equivalences underlying the stochastic process algebras PEPA and IML are related to the bisimilarity of specific FuTSs. By the correspondence result coalgebraic justification of the equivalences of these calculi is obtained. Further illustrations of FuTS semantics are discussed for deterministically (discrete) timed process algebras and Markov Automata.

Contents

1 Introduction 3
2 Preliminaries 6
3 State-to-Function Labelled Transition Systems 8
4 FuTSs coalgebraically 10
5 FuTS-semantics for a elementary process language 17
6 FuTS Semantics of PEPA 23
7 FuTS Semantics of IML 28
8 Discussion 32
9 Concluding remarks 34
1 Introduction

In the last couple of decades, qualitative process description languages have been enriched with quantitative information. In the qualitative case, process description languages equipped with formal operational semantics have proven to be successful formalisms for the modelling of concurrent systems and the analysis of their behaviour. Generally, the operational semantics of a qualitative process description language is given by means of a labelled transition system (LTS), with states being process terms and actions and interactions decorating the transitions between states. Typically, based on the induced transition system relation, a notion of process equivalence is defined, providing means to compare systems and to reduce their representation to enhance subsequent verification.

Extensions of qualitative description languages allowing a stochastic representation of time, usually referred to as stochastic process algebras, or stochastic process calculi (SPCs), are one of the quantitative enrichments of process languages that have received particular attention. For SPCs the main aim has been the integration of qualitative descriptions with quantitative ones in a single mathematical framework, building on the combination of LTSs and continuous-time Markov chains. The latter is one of the most successful approaches to modelling and performance analysis of (computer) systems and networks. An overview of SPCs, equivalences and related analysis techniques can be found in [Hermanns et al.(2002), Baier et al.(2004), Bernardo(2007)], for example. A common feature of many SPCs is that actions are augmented with the rates of exponentially distributed random variables that characterise their duration. Alternatively, actions are assumed to be instantaneous, in which case random variables are used for modelling delays, as in [Hermanns(2002)]. Although exploiting the same class of distributions, the models and techniques underlying the definition of the calculi turn out to be significantly different in many respects. A prominent difference concerns the modelling of the race condition by means of the choice operator, and its relationship to the issue of transition multiplicity. In the quantitative setting, multiplicities can make a crucial distinction between processes that are qualitatively equivalent. Several different approaches have been proposed for handling transition multiplicity. The proposals range from multi-relations [Hillston(1996), Hermanns(2002)], to proved transition systems [Priami(1995)], to LTSs with numbered transitions [van Glabbeek et al.(1995), Hermanns et al.(2002)], to unique rate names [De Nicola et al.(2005)], just to mention a few.

In [De Nicola et al.(2009)], Latella, Massink et al. proposed a variant of LTSs, called Rate Transition Systems (RTSs). In LTSs, a transition is a triple \((P, \alpha, P')\) where \(P\) and \(\alpha\) are the source state and the label of the transition, respectively, while \(P'\) is the target state reached from \(P\) via a transition labelled with \(\alpha\). In RTSs, a transition is a triple of the form \((P, \alpha, \mathcal{P})\). The first and second component are the source state and the label of the transition, as in LTSs, while the third component \(\mathcal{P}\) is a continuation function which associates a non-negative real value to each state \(P'\). A non-zero value for the state \(P'\) represents the rate of the exponential distribution characterising the time for the execution of the action represented by \(\alpha\), necessary to reach \(P'\) from \(P\) via the transition. If \(\mathcal{P}\) maps \(P'\) to 0, then state \(P'\) cannot be reached from \(P\) via this transition with label \(\alpha\). The use of continuation functions provides a clean and simple solution to the transition multiplicity problem and make RTSs particularly suited for SPC semantics. In order to provide a uniform account of the many SPCs proposed in the literature, in previous joint work of the first two authors [De Nicola et al.(2011)] State-to-Function Labelled Transition Systems (FuTSs) have been introduced as a natural generalisation of RTSs. In FuTSs the codomain of the continuation functions are arbitrary semirings, rather than just the non-negative reals. This provides increased flexibility while preserving basic properties of primitive operations like sum and multiplication. Furthermore, FuTSs are equipped with a rich set of (generic) operations on continuation functions, which makes the framework very well suited for the compositional definition of the operational semantics of process calculi, including SPCs and mod-
els where both non-deterministic behaviour and stochastic delays are modelled, like in the Language of Interactive Markov Chains [Hermanns(2002)]. Finally, FuTSs are equipped with a natural notion of bisimilarity which, as we will see, coincides for the concrete cases we studied with the notion of process (strong) equivalences reported in the literature.

In this paper we present a coalgebraic treatment of FuTSs that allows multiple state-to-function transition relations involving arbitrary semirings. Given label sets $\mathcal{L}_i$ and semirings $\mathcal{R}_i$, a FuTS takes the general format $\mathcal{S} = (\mathcal{S}, \langle \rightarrow \rangle_{i=1}^{n})$ with transition relations $\rightarrow \subseteq \mathcal{S} \times \mathcal{L}_i \times \mathcal{F}(\mathcal{S}, \mathcal{R}_i)$. Here, $\mathcal{F}(\mathcal{S}, \mathcal{R}_i)$ are the sets of total functions from $\mathcal{S}$ to $\mathcal{R}_i$ of finite support, a sub-collection of functions also occurring in other work combining coalgebra and quantitative modelling. We will see that the $\mathcal{S}$ is a coalgebra of the product of the functors $\mathcal{F}(\mathcal{S}, \mathcal{R}_i)_{\mathcal{L}_i}$. For this to work, we need the relations $\rightarrow$ to be total and deterministic for the coalgebraic modelling as a function. Maybe surprisingly, this is not a severe restriction at all in the presence of continuation functions: as we will see, the zero-continuation function, which maps every $s'$ to 0 will be associated to a state $s$ and a transition, in order to indicate that no state $s'$ is reachable from $s$ via that transition, in the usual LTS-sense; if $s$ allows a transition to some state $s_1$ as well as to a state $s_2$, then the continuation function will simply yield a non-zero value for $s_1$ and for $s_2$. Therefore, it is no essential limitation to restrict our investigations to total and deterministic FuTSs. For example, by using boolean functions, we can model non-deterministic behaviour, as done in Section 5 and Section 7.

The notion of $\mathcal{S}$-bisimilarity that arises from a FuTS $\mathcal{S}$ is reinterpreted coalgebraically in the present paper. Following a familiar argument, we first prove that the functor associated with a FuTS possesses a final coalgebra and therefore has an associated notion of behavioural equivalence. Then it is shown that behavioural equivalence of the functor induced by $\mathcal{S}$ coincides with bisimilarity for FuTS. Pivotal for the proof is the absence of multiplicities in the FuTS treatment of quantities at the level of the transitions. In fact, quantities are accumulated in the function values of the continuations and hidden at the higher level of abstraction. It is noted, in the presence of a final coalgebra for FuTS a more general definition of behavioural equivalence based on cospans coincides with the one given here, cf. [Kurz(2000)]. The relationship with coalgebraic bisimulation is also investigated and we prove that, under the condition that the underlying semirings admit a (right) multiplicative inverse for non-zero elements, and satisfy the zero-sum property, i.e. a sum in the semiring is equal to zero if and only if all summands are zero, the functors associated to FuTSs preserve weak pullbacks. Consequently, by exploiting a general result on coalgebras, we get that for these functors behavioural equivalence coincides with coalgebraic bisimilarity.

Using the bridge established by the correspondence results, we continue by showing for two well-known stochastic process algebras, viz. Hillston’s PEPA [Hillston(1996)] and Hermanns’s IML [Hermanns(2002)], that their standard notion of strong equivalence and strong bisimilarity coincide with bisimilarity of the associated FuTS (and thus with behavioural equivalence and coalgebraic bisimilarity of the corresponding functor). PEPA stands out as one of the prominent Markovian process algebras, and IML specifically provides separate prefix constructions for actions and for delays. The equivalences of PEPA and of IML are compared with the bisimulations of the respective FuTSs as given by an alternative operational semantics involving the state-to-function scheme. In passing, the multiplicities have to be dealt with. Appropriate lemmas are provided relating the relation-based cumulative treatment with FuTSs to the multi-relation-based explicit treatment of PEPA and IML. It is noted that in our treatment below we restrict to the key-fragment of these two SPCs.

We finally discuss how FuTSs can be used also for the definition of the semantics of deterministically discrete timed process algebras as well as for models which incorporate at the same time non-determinism, discrete probabilities and Markovian randomised delays, like it is the case in Markov
Automata as presented in [Eisentraut et al.(2010a), Eisentraut et al.(2010b)].

Related work on coalgebra includes the papers [de Vink & Rutten(1999), Klin & Sassone(2008)] and [Sokolova(2011)]. These papers also cover measures and congruence formats, a topic not touched upon here. For the discrete parts, regarding the correspondence of bisimulations, our work aligns with the approach of the papers mentioned. In this paper the bialgebraic perspective of SOS and bisimulation [Turi & Plotkin(1997)] is left implicit. An interesting direction of research combining coalgebra and quantities studies various types of weighted automata, including linear weighted automata, and associated notions of bisimulation and languages, as well as algorithms for these notions [Boreale(2009), Klin(2009), Silva et al.(2011), Bonchi et al.(2011)]. Klin considers weighted transition systems, labeled transition systems that assign a weight to each transition. For commutative monoids the notion of a weighted transition system compares with our notion of a FuTS, and for which, when casted to the coalgebraic setting, the associated concept of bisimulation coincides with observational equivalence. Building on a result on bounded functors [Gumm & Schröder(2001)], it is shown in [Bonchi et al.(2011)] for a functor involving functions of finite support over a field that the final coalgebra exists. In the present paper, we have followed the scheme of [Bonchi et al.(2011)] to obtain such a result for a functor induced by a FuTS. The process languages with stochastic delays we consider in the sequel, based on PEPA and IML, involve a multi-way CSP-like parallel operator; components proceed simultaneously when synchronization on an action from the synchronization alphabet is possible. However, here we do not distinguish between internal and external non-determinism, cf. [Hoare(1985)], as an explicit representation of such a distinction is neither contemplated in PEPA, nor in IML. A coalgebraic treatment of this distinction is proposed in [Wolter(2002)], which uses a functor for so-called non-deterministic filter automata, viz. \( \mathcal{P}(\mathcal{P}(\mathcal{A})) \times [\mathcal{A} \rightarrow \mathcal{P}(\cdot)] \) involving partial functions from a set of actions \( \mathcal{A} \) to a finite power-set. Via currying, this can be brought in the form \( \mathcal{FS}(\cdot, \mathbb{B}) \) for \( \mathcal{L} = \mathcal{P}(\mathcal{P}(\mathcal{A})) \times \mathcal{A} \), fitting in the format of the functor for the FuTSs considered here. In [Boreale & Gadducci(2006)] processes are interpreted as formal power-series over a semiring in the style of [Rutten(2003)]. This allows to compare testing equivalence for a CSP-style language and bisimulation in a Moore automaton. It is noted, that the notions of equivalence addressed in this paper, as often in coalgebraic treatments of process relations, are all strong bisimilarities.

Structure of the paper The present paper is organised as follows: Section 2 briefly discusses some material on semirings and coalgebras. FuTSs as well as the associated notion of bisimulation are presented in Section 3. The coalgebraic counterparts of FuTSs and FuTS-bisimilarity are defined in Section 4 where we also establish the correspondence with behavioural equivalence of the final coalgebra and with coalgebraic bisimilarity. As a stepping stone towards the treatment of PEPA and IML, we discuss in Section 5 an elementary process language that constitutes the qualitative core of the two SPCs. In Section 6 the standard equivalence of PEPA is identified with the bisimulation of a FuTS and, hence, with behavioural equivalence and coalgebraic bisimilarity. In Section 7 the same is done for the language of IMCs where actions and delays are present on equal footing. A discussion of our results and possible extensions is presented in Section 8. Finally, Section 9 wraps up and discusses directions of future research.

An extended abstract of part of this manuscript has appeared as [Latella et al.(2012)]. The additional contributions of the present paper include a detailed proof of the existence of the final coalgebra of the relevant functors and the investigation on the relationship between FuTS bisimilarity and behavioural equivalence, on one side, and coalgebraic bisimilarity, on the other. In particular, it is shown that the functor type involved preserves weak pullbacks when the underlying semiring, amongst other, satisfy the zero-sum property. As illustration we insert the modelling of a basic qualitative process lan-
guage with \textit{FuTSs}. Finally, we provide a discussion of the \textit{FuTS}-based approach and its coalgebraic view to deterministically discrete timed process algebras and Markov Automata.

## 2 Preliminaries

A tuple $\mathcal{R} = (\mathcal{R}, +, 0, \cdot, 1)$ is called a semiring, if $(\mathcal{R}, +, 0)$ is a commutative monoid with neutral element $0$, $(\mathcal{R}, \cdot, 1)$ is a monoid with neutral element $1$, $\cdot$ distributes over $+$, and $0 \cdot r = r \cdot 0 = 0$ for all $r \in \mathcal{R}$. As examples of a semiring we will use the booleans $\mathbb{B} = \{ \text{false}, \text{true} \}$ with disjunction as sum and conjunction as multiplication, and the non-negative reals $\mathbb{R}_{\geq 0}$ with the standard operations.

We will consider, for a semiring $\mathcal{R}$ and a function $\varphi : X \to \mathcal{R}$, countable sums $\sum_{x \in X} \varphi(x)$ in $\mathcal{R}$, for $X' \subseteq X$. For such a sum to exist we require $\varphi$ to be of finite support, i.e. the support set $\text{spt}(\varphi) = \{ x \in X | \varphi(x) \neq 0 \}$ is finite. Finally, for $\varphi : X \to \mathcal{R}$, and $X' \subseteq X$, we let $\varphi[X'] = \{ \varphi(x) | x \in X' \}$.

We use the notation $\mathcal{FS}(X, \mathcal{R})$ for the collection of all functions of finite support from the set $X$ to the semiring $\mathcal{R}$. A construct $[x_1 \mapsto r_1, \ldots, x_n \mapsto r_n]$, with, for $i = 1 \ldots n$, $x_i \in X$ all distinct and $r_i \in \mathcal{R}$, denotes the function that assigns $r_i$ to $x_i$, $i = 1 \ldots n$, and assigns $0$ to all $x \in X$ different from all $x_i$. In particular $[]$, or more precisely $[\ ]_{\mathcal{R}}$, is the constant function $x \mapsto 0$ and $\mathcal{D}_{\mathcal{R}, x} = [x \mapsto 1]$ is the Dirac function on $\mathcal{R}$ for $x \in X$; in the sequel we will often drop the subscript $\mathcal{R}$ from $[]_{\mathcal{R}}$ and $\mathcal{D}_{\mathcal{R}, x}$, when the semiring is clear from the context.

We use $\oplus \varphi$ for the value $\sum_{x \in X} \varphi(x)$ in $\mathcal{R}$. For $\varphi, \psi \in \mathcal{FS}(X, \mathcal{R})$, the function $\varphi + \psi$ is the pointwise sum of $\varphi$ and $\psi$, i.e. $\varphi + \psi(x) = \varphi(x) + \psi(x) \in \mathcal{R}$. Clearly, $\varphi + \psi$ is of finite support as $\varphi$ and $\psi$ are. Given an injective operation $| : X \times X \to X$, we define $\varphi | \psi : X \to \mathcal{R}$, by $(\varphi | \psi)(x) = \varphi(x_1) \cdot \psi(x_2)$ if $x = x_1 \cdot x_2$ for some $x_1, x_2 \in X$, and $(\varphi | \psi)(x) = 0$ otherwise. Injectivity of the operation $|$ guarantees that $\varphi | \psi$ is well-defined. Again, $\varphi | \psi$ is of finite support as $\varphi$ and $\psi$ are. This is used in the setting of syntactic processes $P$ that may have the form $P_1 | P_2$ for two processes $P_1$ and $P_2$ and a syntactic operator $|$. We have the following properties.

**Lemma 1** Let $X$ be a set, $\mathcal{R}$ a semiring, and $| \ $ an injective binary operation on $X$. For $\varphi, \psi \in \mathcal{FS}(X, \mathcal{R})$ it holds that $\oplus(\varphi + \psi) = \oplus \varphi + \oplus \psi$ and $\oplus(\varphi | \psi) = (\oplus \varphi) \cdot (\oplus \psi)$.

We recall some basic definitions from coalgebra. See e.g. \cite{Rutten2000} for more details. For a functor $\mathcal{F} : \text{Set} \to \text{Set}$ on the category $\text{Set}$ of sets and functions, a coalgebra of $\mathcal{F}$ is a set $X$ together with a mapping $\alpha : X \to \mathcal{F}(X)$. A homomorphism between two $\mathcal{F}$-coalgebras $(X, \alpha)$ and $(Y, \beta)$ is a function $f : X \to Y$ such that $\mathcal{F}(f) \circ \alpha = \beta \circ f$. See Figure 1. An $\mathcal{F}$-coalgebra $(\Omega_\mathcal{F}, \omega_\mathcal{F})$ is called final, if there exists, for every $\mathcal{F}$-coalgebra $S = (X, \alpha)$, a unique homomorphism $\| \|_\mathcal{F} : (X, \alpha) \to (\Omega_\mathcal{F}, \omega_\mathcal{F})$. Two elements $x_1, x_2$ of a $\mathcal{F}$-coalgebra $S = (X, \alpha)$ are called behavioural equivalent with respect to $\mathcal{F}$ if $\| x_1 \|_\mathcal{F} = \| x_2 \|_\mathcal{F}$, denoted $x_1 \approx_\mathcal{F} x_2$. In the notation $\| \|_\mathcal{F}$ as well as $\approx_\mathcal{F}$, the indication of the specific coalgebra $S$ will be omitted, when clear from the context.

Using a characterisation of \cite{GummSchroder02}, a functor $\mathcal{F}$ on $\text{Set}$ is bounded, if there exist sets $A$ and $B$ and a surjective natural transformation $\eta : A \times (\cdot)^B \Rightarrow \mathcal{F}$. Here, $A \times (\cdot)$ is the functor that maps a set $X$ to the Cartesian product $A \times X$ and maps a function $f : X \to Y$ to the mapping $A \times f : A \times X \to A \times Y$ with $(a, f(x)) = (a, f(x))$, while $(\cdot)^B$ denotes the functor that maps a set $X$ to the function space $X^B$ of all functions from $B$ to $X$ and that maps a function $f : X \to Y$ to the mapping $f^B : X^B \to Y^B$ with $f^B(\varphi)(b) = f(\varphi(b))$.

In order to deal with product functors in the sequel, we will use the following lemma, where $B_1 + \cdots + B_n$ is the disjoint union of $B_1, \ldots, B_n$ and $\uparrow$ denotes restriction on functions.
Let $A_1,\ldots,A_n, B_1,\ldots,B_n$ be sets and $\mathcal{F}_1,\ldots,\mathcal{F}_n$ be functors on Set. Suppose $\eta^i : A_i \times (\cdot)^{B_i} \Rightarrow \mathcal{F}_i$ is a natural transformation such that $\eta^i_X : A_i \times X^{B_i} \to \mathcal{F}_i(X)$, $i = 1\ldots,n$, is surjective. Then $\eta : A_1 \times \cdots \times A_n \times (\cdot)^{B_1+\cdots+B_n} \Rightarrow \mathcal{F}_1 \times \cdots \times \mathcal{F}_n$ with $\eta_X : A_1 \times \cdots \times A_n \times (X)^{B_1+\cdots+B_n} \to \mathcal{F}_1(X) \times \cdots \times \mathcal{F}_n(X)$ such that

$$\eta_X((a_1,\ldots,a_n), \varphi) = (\eta^1_X(a_1, \varphi|B_1), \ldots, \eta^n_X(a_n, \varphi|B_n))$$

is a natural transformation. Moreover, for each set $X$, the mapping $\eta_X$ is surjective. $\square$

The lemma can be straightforwardly checked. Thus, a product functor $\mathcal{F}_1 \times \cdots \times \mathcal{F}_n$ on Set, for which each factor $\mathcal{F}_i$, $i = 1\ldots,n$, meets the criterion of [Gumm & Schröder(2002)], also meets the criterion itself and hence $\mathcal{F}_1 \times \cdots \times \mathcal{F}_n$ is bounded.

For bounded functors we have the following result, see [Gumm & Schröder(2001)] for a proof.

**Theorem 3** If a functor $\mathcal{F} : \text{Set} \to \text{Set}$ is bounded, then its final coalgebra exists. $\square$

An $\mathcal{F}$-coalgebra $(R, \gamma)$ with $R \subseteq X \times Y$ is called a coalgebra bisimulation of two $\mathcal{F}$-coalgebras $(X, \alpha)$ and $(Y, \beta)$ if $\alpha \circ \pi_1 = \mathcal{F}(\pi_1) \circ \gamma$ and $\beta \circ \pi_2 = \mathcal{F}(\pi_2) \circ \gamma$. Here $\pi_1 : R \to X$ and $\pi_2 : R \to Y$ are the projections from $R$ to $X$ and $Y$, respectively. See Figure 1. For a coalgebra $S = (X, \alpha)$, two elements $x_1, x_2 \in X$ are called coalgebraically bisimilar if there exists a coalgebra bisimulation $R$ of $(X, \alpha)$ and $(X, \alpha)$ such that $R(x_1, x_2)$, notation $x_1 \sim_{\mathcal{F}} x_2$, or simply $x_1 \sim x_2$ when the coalgebra $S$ is clear from the context.

Given a cospan $f_1 : A \to C$ and $f_2 : B \to C$, their pullback in Set is a set $P$ together with a span $p_1 : P \to A$ and $p_2 : P \to B$ such that $f_1 \circ p_1 = f_2 \circ p_2$, and, moreover, for any $Q$ and span $q_1 : Q \to A$ and $q_2 : Q \to B$ satisfying $f_1 \circ q_1 = f_2 \circ q_2$ there exists a unique mapping $m : Q \to P$, called the mediating morphism, such that $q_1 = p_1 \circ m$ and $q_2 = p_2 \circ m$. In case the mediating morphism need not to be unique, we speak of a weak pullback of $f_1 : A \to C$ and $f_2 : B \to C$. In Set, the pullback of $f_1 : A \to C$ and $f_2 : B \to C$ is the set $P = \{(a, b) \in A \times B \mid f_1(a) = f_2(b)\}$ together with the projections $p_1 : P \to A$ and $p_2 : P \to B$.

If a functor $\mathcal{F}$ transforms a weak pullback diagram for $f : A \to C$ and $g : B \to C$ into a weak pullback diagram of $\mathcal{F}(f) : \mathcal{F}(A) \to \mathcal{F}(C)$, and $\mathcal{F}(g) : \mathcal{F}(B) \to \mathcal{F}(C)$, the functor $\mathcal{F}$ is said to preserve weak pullbacks. More explicitly, $\mathcal{F}$ preserves weak pullbacks, if for any cospan $f : A \to C$ and $g : B \to C$ in Set, having a weak pullback $W$ together with span $w_1$ and $w_2$, we have that $\mathcal{F}(f) \circ \mathcal{F}(w_1) = \mathcal{F}(f) \circ \mathcal{F}(w_2)$, and, for any set $Q$ and span $F : Q \to \mathcal{F}(A)$ and $G : Q \to \mathcal{F}(B)$ with $\mathcal{F}(f) \circ F = \mathcal{F}(g) \circ G$, we can find a mediating morphism $M : Q \to \mathcal{F}(W)$ such that $F = \mathcal{F}(w_1) \circ M$ and $G = \mathcal{F}(w_2) \circ M$. See Figure 2.
In \textit{Set}, for many functors coalgebraic bisimulation and behavioural equivalence coincide (but not always, see [Bonchi et al.(2011), Section 2.2] for an example). A sufficient condition for the two notions being equal is the preservation of weak pullbacks, which is for many functors the case (but not all, in particular not for the Giry-functor on the category of measurable spaces and functions, see [Viglizzo(2005), Marti & Venema(2012)]).

\textbf{Theorem 4} [Rutten(2000), Theorem 9.3] If a functor \( \mathcal{F} \) on \( \textit{Set} \) preserves weak pullbacks, then coalgebraic bisimilarity and behavioural equivalence coincide, i.e. for any \( \mathcal{F} \)-coalgebra \( S = (X, \alpha) \) and elements \( x_1, x_2 \in X \) it holds that \( x_1 \sim^S_{\mathcal{F}} x_2 \) iff \( x_1 \approx^S_{\mathcal{F}} x_2 \). \hfill \Box

A number of proofs of results on process languages \( \mathcal{P} \) in this paper rely on so-called guarded recursion [de Bakker & de Vink(1996)]. Typically, constants \( X \), also called process variables, are a syntactical ingredient in these languages. As usual, if \( X \) is the body, we require \( P \) to be prefix-guarded, i.e. any occurrence of a constant in the body \( P \) is in the scope of a prefix-construct of the language. Guarded recursion assumes the existence of a function \( c : \mathcal{P} \rightarrow \mathbb{N} \) such that \( c(P) = 1 \) if \( P \) is a prefix construct, \( c(P_1 \bullet P_2) > \max\{ c(P_1), c(P_2) \} \) for all other syntactic operators \( \bullet \) of \( \mathcal{P} \), and moreover \( c(X) > c(P) \) if \( X \) is the body.

\section{State-to-Function Labelled Transition Systems}

The definition of a state-to-function labelled transition system, \textit{FuTS} for short, involves a set \( S \) of states and one or more relations of states on the one hand, and functions from states into semirings on the other hand. For sums over arbitrary subsets of states to exist, the functions are assumed to be of finite support.

\textbf{Definition 1} A \textit{FuTS} \( S \), in full 'a state-to-function labelled transition system', over a number of label sets \( L_i \) and semirings \( R_i \), \( i = 1 \ldots n \), is a tuple \( S = (S, (\rightarrow_i)_{i=1}^n) \) such that, for \( i = 1 \ldots n \), \( \rightarrow_i \subseteq S \times L_i \times \mathcal{F}(S, R_i) \).

Similar as for state-to-state transitions of \textit{LTS}s, for state-to-function transitions of \textit{FuTS}s we write \( s \xrightarrow{\ell} v \) for \( (s, \ell, v) \in \rightarrow_i \). For a \textit{FuTS} \( S = (S, (\rightarrow_i)_{i=1}^n) \) the set \( S \) is called the set of states or the carrier set. We refer to each \( \rightarrow_i \) as a state-to-function transition relation of \( S \) or just as a transition relation. If for \( S \) we have that \( n = 1 \), i.e. there is only one state-to-function transition relation \( \rightarrow \),
then $S$ is called simple. A $FuTS$ $S$ is called total and deterministic if for each transition relation $\rightarrow_i \subseteq S \times L_i \times FS(S, R_i)$ involved and for all $s \in S$, $i \in L_i$, we have $s \not\rightarrow_i v$ for exactly one $v \in FS(S, R_i)$. In such a situation, the state-to-function relations $\rightarrow_i$ correspond to functions $S \rightarrow L_i \rightarrow FS(S, R_i)$. For the remainder of the paper, all $FuTS$s we consider will be total and deterministic, unless explicitly stated otherwise. It is noted that Definition 1 slightly differs in formulation from the one provided in [De Nicola et al.(2011)].

**Examples** For the modeling of standard interactive processes with $FuTS$s, we choose a set of actions $A$ as label set and the booleans $\mathbb{B}$ as semiring. Consider the two processes $P = a.b.nil + a.c.nil$ and $Q = a.(b.nil + c.nil)$, their representation as a $FuTS$ is depicted in Figure 3. For process $P$, on the left of the figure, we have $P \rightarrow_i [b.nil \mapsto\text{true}, c.nil \mapsto\text{true}]$, while for the process $Q$, in the middle of the figure, we have $Q \rightarrow_i [b.nil + c.nil \mapsto\text{true}]$. So, for $P$ the two processes $b.nil$ and $c.nil$ each are set to $\text{true}$ and for $Q$ only the process $b.nil + c.nil$ is set to $\text{true}$. Since any finite number of alternatives can be assigned a non-zero value by a function of finite support, deterministic $FuTS$s are able to represent image-finite non-determinism; the branching is taken care of by the functions from process terms to $\mathbb{B}$. To complete the picture $b.nil \rightarrow [\text{nil} \mapsto\text{true}], c.nil \rightarrow [\text{nil} \mapsto\text{true}], b.nil + c.nil \rightarrow [\text{nil} \mapsto\text{true}]$ and $b.nil + c.nil \rightarrow [\text{nil} \mapsto\text{true}]$.

As another example of a simple $FuTS$, Figure 3 displays at its right a $FuTS$ over the action set $A$ and the semiring $\mathbb{R}_{\geq 0}$ of the non-negative real numbers. The functions $v_0$ to $v_3$ used in the example have the property that $\oplus v_i(s) = 1$, for $i = 0 \ldots 3$. More explicitly we have

\[
\begin{align*}
 s_0 &\rightarrow [s_0 \mapsto \frac{1}{2}, s_1 \mapsto \frac{1}{2}] \\
 s_2 &\rightarrow [s_2 \mapsto \frac{1}{2}, s_3 \mapsto \frac{1}{2}] \\
 s_3 &\rightarrow [s_0 \mapsto \frac{1}{2}, s_3 \mapsto \frac{1}{2}] \\
 s_1 &\rightarrow [s_1 \mapsto \frac{1}{2}, s_2 \mapsto \frac{1}{2}] \\
 s_i &\rightarrow [\mathbb{B}] \text{ for } i = 0, 2, 3
\end{align*}
\]

Usually, such a $FuTS$ over $\mathbb{R}_{\geq 0}$ is called a (reactive) probabilistic transition system, using standard terminology introduced in [van Glabbeek et al.(1995)].

In Section 7 we will provide semantics for the process language $IML$ for interactive Markov chains [Hermanns(2002)] [Hermanns & Katoen(2010)] using $FuTS$s. Unlike many other stochastic process algebras, a single $IML$ process can in general both perform action-based transitions and time-delays governed by exponential distributions.
Below it will be notationally convenient to consider a (total and deterministic) FuTS as a tuple $(S, (\theta_i)_{i=1}^n)$ with transition functions $\theta_i : S \to L_i \to \mathcal{FS}(S, \mathcal{R}_i)$, $i = 1 \ldots n$, rather than using the form $(S, (\to_s)_{i=1}^n)$ that occurs more frequently for concrete examples in the literature. Alternatively, using disjoint unions, one could see a FuTS represented by a function $\theta'$ of type $S \to \bigoplus_{i=1}^n L_i \to \bigoplus_{i=1}^n \mathcal{FS}(S, \mathcal{R}_i)$ satisfying the additional property that $\theta'(s)(\ell) \in \mathcal{FS}(S, \mathcal{R}_i)$ if $\ell \in L_i$. As this fits less smoothly with the category-theoretical approach of Section 4, we stick to the former format.

We will use the notation with transition functions $\theta_i : S \to L_i \to \mathcal{FS}(S, \mathcal{R}_i)$ to introduce the notion of bisimilarity for a FuTS.

**Definition 2** Let $S = (S, (\theta_i)_{i=1}^n)$ be a FuTS over the label sets $L_i$ and semirings $\mathcal{R}_i$, $i = 1 \ldots n$. An equivalence relation $R \subseteq S \times S$ is called an $\mathcal{S}$-bisimulation if $R(s_1, s_2)$ implies

$$\sum_{r \in [t]_{R_\theta}} \theta_1(s_1)(\ell)(t') = \sum_{r \in [t]_{R_\theta}} \theta_2(s_2)(\ell)(t')$$

for all $t \in S$, $i = 1 \ldots n$ and $\ell \in L_i$, where we use the notation $[t]_{R_\theta}$ to denote the equivalence class of $t$ in $S$ with respect to $R$. Two elements $s_1, s_2 \in S$ are called $\mathcal{S}$-bisimilar if $R(s_1, s_2)$ for some $\mathcal{S}$-bisimulation $R$ for $S$. Notation $x_1 \approx_S x_2$.

Note that the sums in equation $[1]$ exist since the functions $\theta_i(s_1)(\ell), \theta_i(s_2)(\ell) \in \mathcal{FS}(S, \mathcal{R}_i)$, $i = 1 \ldots n$, are of finite support.

For the combined FuTS of the two processes $P = a.b.nil + a.c.nil$ and $Q = a.(b.nil + c.nil)$ of Figure 3 consider the equivalence relation $R$ such that $R(P, Q)$ and also $R(b.nil, b.nil + c.nil)$, $R(b.nil, b.nil + c.nil)$, $R(c.nil b.nil + c.nil)$, and $R(nil, nil)$. Then $R$ is not a FuTS-bisimulation. Although, on the one hand, $\sum_{r \in [nil]_{R_\theta}} \theta(b.nil)(b)(t') = \theta(b.nil)(b)(nil) = \text{true}$ and, on the other hand, $\sum_{r \in [nil]_{R_\theta}} \theta(b.nil + c.nil)(b)(t') = \theta(b.nil + c.nil)(b)(nil) = \text{true}$, we have $\sum_{r \in [nil]_{R_\theta}} \theta(b.nil)(c)(t') = \text{false}$, while $\sum_{r \in [nil]_{R_\theta}} \theta(b.nil + c.nil)(c)(t') = \text{true}$, taking sums, i.e. disjunctions, in $\mathbb{B}$. As no other equivalence relation fulfills the requirements of Definition 2 either, we conclude that the processes $P$ and $Q$ are not bisimilar. See Section 5 for more detail.

For FuTSs that are neither total nor deterministic a variation of Definition 2 applies involving the usual transfer conditions: if $R(s_1, s_2)$ then

$$s_1 \overset{\ell_i}{\rightarrow} v_1 \in \mathcal{FS}(S, \mathcal{R}_i) \implies \exists v_2 : s_2 \overset{\ell_i}{\rightarrow} v_2 \in \mathcal{FS}(S, \mathcal{R}_i) \land R(v_1, v_2)$$

$$s_2 \overset{\ell_i}{\rightarrow} v_2 \in \mathcal{FS}(S, \mathcal{R}_i) \implies \exists v_1 : s_1 \overset{\ell_i}{\rightarrow} v_1 \in \mathcal{FS}(S, \mathcal{R}_i) \land R(v_1, v_2)$$

where the lifting $R$ on $\mathcal{FS}(S, \mathcal{R}_i) \times \mathcal{FS}(S, \mathcal{R}_i)$ is given by

$$R(v_1, v_2) \iff \sum_{r \in [t]_{R_\theta}} v_1(t') = \sum_{r \in [t]_{R_\theta}} v_2(t')$$

As we will consider total and deterministic FuTSs only, we stick to the more convenient formulation involving transition functions of Definition 2.

4 FuTSs coalgebraically

In this section we will cast FuTSs in the framework of coalgebras and prove a correspondence result of FuTS-bisimulation and behavioural equivalence for a suitable functor on Set. We also show that
for the functors associated with FuTSs, under mild conditions for the semirings involved, behavioural equivalence and coalgebraic bisimulation coincide.

**Definition 3** Let \( \mathcal{L} \) be a set of labels and \( \mathcal{R} \) a semiring. The functor \( \mathcal{U}_\mathcal{L}^\mathcal{R} : \text{Set} \to \text{Set} \) assigns to a set \( X \) the function space \( \mathcal{F}\mathcal{S}(X, \mathcal{R})^\mathcal{L} \) of all functions \( \varphi : \mathcal{L} \to \mathcal{F}\mathcal{S}(X, \mathcal{R}) \) and assigns to a mapping \( f : X \to Y \) the mapping \( \mathcal{U}_\mathcal{L}^\mathcal{R}(f) : \mathcal{F}\mathcal{S}(X, \mathcal{R})^\mathcal{L} \to \mathcal{F}\mathcal{S}(Y, \mathcal{R})^\mathcal{L} \) where

\[
\mathcal{U}_\mathcal{L}^\mathcal{R}(f)(\varphi)(y) = \sum_{x \in f^{-1}(y)} \varphi(\ell)(x)
\]

for all \( \varphi \in \mathcal{F}\mathcal{S}(X, \mathcal{R})^\mathcal{L}, \ell \in \mathcal{L} \) and \( y \in Y \).

Again we rely on \( \varphi(\ell) \in \mathcal{F}\mathcal{S}(X, \mathcal{R}) \) having a finite support for the sum to exist and for \( \mathcal{U}_\mathcal{L}^\mathcal{R} \) to be well-defined. In fact, we have \( \text{spt}(\mathcal{U}_\mathcal{L}^\mathcal{R}(f)(\varphi)(\ell)) \subseteq \{ f(x) \mid x \in \text{spt}(\varphi(\ell)) \} \). We furthermore observe that for any simple FuTS \((S, \theta)\) over \( \mathcal{L} \) and \( \mathcal{R} \) we have \( \theta : S \to \mathcal{L} \to \mathcal{F}\mathcal{S}(S, \mathcal{R}) \). Thus \((S, \theta)\) can be interpreted as a \( \mathcal{U}_\mathcal{L}^\mathcal{R} \)-coalgebra. In the sequel, we will abbreviate \( \mathcal{U}_\mathcal{L}^\mathcal{R} \) with \( \mathcal{U} \) whenever \( \mathcal{L} \) and \( \mathcal{R} \) are clear from the context.

As we aim to compare our notion of bisimulation for FuTSs with behavioural equivalence for the functor \( \mathcal{U} \), given a set of labels \( \mathcal{L} \) and a semiring \( \mathcal{R} \), we need to check that \( \mathcal{U} \) possesses a final coalgebra. For this, we adapt the proof for the functor \( \mathcal{F}\mathcal{S}(\cdot, \mathcal{M}) : \text{Set} \to \text{Set} \) where \( \mathcal{M} \) is a monoid (rather than a semiring) as sketched in [Silva(2010)] to the setting here. The proof exploits the characterisation of \([\text{Gumm and Schröder}(2002)]\), see Section 2. An alternative route to showing the existence of a final coalgebra is followed in [Klin(2009)], also for commutative monoids, and relies on a finitary result of Barr, cf. [Barr(1993)].

**Lemma 5** Let \( \mathcal{L} \) be a set of labels, \( \mathcal{R} \) a semiring. Then the functor \( \mathcal{U} \) on \( \text{Set} \) is bounded.

**Proof** We verify that \( \eta : \mathcal{F}\mathcal{S}(\mathcal{N}, \mathcal{R})^\mathcal{L} \times (\cdot)^{\mathcal{L} \times \mathcal{N}} \rightarrow \mathcal{F}\mathcal{S}(\cdot, \mathcal{R})^\mathcal{L} \), defined by \( \eta_X : \mathcal{F}\mathcal{S}(\mathcal{N}, \mathcal{R})^\mathcal{L} \times \mathcal{X}^{\mathcal{L} \times \mathcal{N}} \rightarrow \mathcal{F}\mathcal{S}(X, \mathcal{R})^\mathcal{L} \) such that \( \eta_X(v, \xi)(\ell)(x) = \sum_{(\ell, n) \in \xi} \nu(\ell)(n) \), is a natural transformation with surjective components. Note that \( \eta \) is of the form \( A \times (\cdot)^{\mathcal{B}} \) for the sets \( A = \mathcal{F}\mathcal{S}(\mathcal{N}, \mathcal{R})^\mathcal{L} \) and \( B = \mathcal{L} \times \mathcal{N} \) as the characterisation requires.

For \( f : X \to Y \) we check that the diagram

\[
\begin{array}{ccc}
\mathcal{F}\mathcal{S}(\mathcal{N}, \mathcal{R})^\mathcal{L} \times \mathcal{X}^{\mathcal{L} \times \mathcal{N}} & \xrightarrow{id \times f} & \mathcal{F}\mathcal{S}(\mathcal{N}, \mathcal{R})^\mathcal{L} \times \mathcal{Y}^{\mathcal{L} \times \mathcal{N}} \\
\eta_X & & \eta_Y \\
\mathcal{F}\mathcal{S}(X, \mathcal{R})^\mathcal{L} \times \mathcal{F}\mathcal{S}(f, \mathcal{R})^\mathcal{L} \times \mathcal{F}\mathcal{S}(f, \mathcal{R})^\mathcal{L} & \xrightarrow{ \eta(f) } & \mathcal{F}\mathcal{S}(Y, \mathcal{R})^\mathcal{L}
\end{array}
\]

commutes, where \( f^{\mathcal{L} \times \mathcal{N}} : \mathcal{X}^{\mathcal{L} \times \mathcal{N}} \rightarrow \mathcal{Y}^{\mathcal{L} \times \mathcal{N}} \) is, as usual, given by \( f^{\mathcal{L} \times \mathcal{N}}(\xi) = f \circ \xi \). We have, for \( v \in \mathcal{F}\mathcal{S}(\mathcal{N}, \mathcal{R})^\mathcal{L}, \xi \in \mathcal{X}^{\mathcal{L} \times \mathcal{N}}, \ell \in \mathcal{L} \) and \( y \in Y \), that

\[
\begin{align*}
\mathcal{F}\mathcal{S}(f, \mathcal{R})^\mathcal{L}(\eta_X(v, \xi))(\ell)(y) & = \sum_{x \in f^{-1}(y)} \eta_X(v, \xi)(\ell)(x) & \text{definition } \mathcal{F}\mathcal{S}(f, \mathcal{R})^\mathcal{L} \\
& = \sum_{x \in f^{-1}(y)} \sum_{(\ell, n) \in \xi} \nu(\ell)(n) & \text{definition } \eta_X \\
& = \sum_{(\ell, n) \in (f \circ \xi)^{-1}(y)} \nu(\ell)(n) & (f \circ \xi)^{-1}(y) = \xi^{-1}(f^{-1}(y)) \\
& = \eta_Y(v, (f \circ \xi))(\ell)(y) & \text{definition } \eta_Y \\
& = \eta_Y((id \times f^{\mathcal{L} \times \mathcal{N}})(v, \xi))(\ell)(y) & \text{definitions of } id_{\mathcal{F}\mathcal{S}(\mathcal{N}, \mathcal{R})^\mathcal{L}} \text{ and } f^{\mathcal{L} \times \mathcal{N}}
\end{align*}
\]
Thus, we have that \( \mathcal{F}S(f, R)^L(\eta_X(\nu, \xi)) = \eta_Y((id_{\mathcal{F}S(X, R)^L} \times f^{\mathcal{F}S(X, R)^L})(\nu, \xi)) \) and \( \mathcal{F}S(f, R)^L \circ \eta_X = \eta_Y \circ (id_{\mathcal{F}S(X, R)^L} \times f^{\mathcal{F}S(X, R)^L}) \).

As to the surjectivity of \( \nu_X : \mathcal{F}S(\mathbb{N}, R)^L \times X^{\mathcal{F}S(\mathbb{N}, R)^L} \to \mathcal{F}S(X, R)^L \), for any set \( X \). Choose \( \varphi \in \mathcal{F}S(\mathbb{N}, R)^L \). Pick \( \ell \in L \). Suppose \( \text{spt}(\varphi(\ell)) = \{x_1, \ldots, x_s\} \subseteq X \). Pick \( x_0 \in X \) arbitrary. We define \( \nu_{\varphi} \in \mathcal{F}S(\mathbb{N}, R)^L \) and \( \xi : L \times \mathbb{N} \rightarrow X \) as follows:

\[
\nu_{\varphi}(\ell)(i) = \begin{cases} 
\varphi(\ell)(x_i) & \text{for } i = 1 \ldots s \\
0 & \text{otherwise}
\end{cases}
\text{ and } \xi_{\varphi}(\ell, i) = \begin{cases} 
x_i & \text{for } i = 1 \ldots s \\
x_0 & \text{otherwise}
\end{cases}
\]

Note, \( \nu_{\varphi}(\ell) \) has finite support, for \( \ell \in L \). Then we have

\[
\eta_X(\nu_{\varphi}, \xi_{\varphi})(\ell)(x) = \sum \langle (\ell, n) \in \xi_{\varphi}^{-1}(x) \rangle \nu_{\varphi}(\ell)(n) \quad \text{definition } \eta_X
\]

\[
= \sum \langle (\ell, i) \in \xi_{\varphi}^{-1}(\ell, x), x_i \in L \rangle \nu_{\varphi}(\ell)(i)
\]

\[
= \begin{cases} 
\varphi(\ell)(x_i) & \text{if } x = x_i, i = 1 \ldots s \\
0 & \text{otherwise}
\end{cases}
\text{ sums of 0 or 1 summand}
\]

\[
= \varphi(\ell)(x)
\text{ spt}(\varphi(\ell)) = \{x_1, \ldots, x_s\}
\]

Hence \( \eta_X(\nu_{\varphi}, \xi_{\varphi}) = \varphi \) and \( \eta_X \) is surjective.

Working with total and deterministic FuTSs, we can interpret a FuTS \( S = (S, (\theta_i)_{i=1}^n) \) over the label sets \( L_i \) and semirings \( R_i, i = 1 \ldots n \), as a product \( \theta_1 \times \cdots \times \theta_n : S \rightarrow \prod_{i=1}^n (L_i \rightarrow \mathcal{F}S(S, R_i)) \) of functions \( \theta_i : S \rightarrow L_i \rightarrow \mathcal{F}S(S, R_i) \). To push this idea a bit further, we want to consider the FuTS \( S = (S, (\theta_i)_{i=1}^n) \) as a coalgebra of a suitable product functor on \( \text{Set} \).

**Definition 4** Let \( L = (L_1, \ldots, L_n) \) be an n-tuple of label sets and \( R = (R_1, \ldots, R_n) \) be an n-tuple of semirings. The functor \( \mathcal{V}_R^L \) on \( \text{Set} \) is defined by \( \mathcal{V}_R^L = \prod_{i=1}^n \mathcal{U}_{\mathcal{V}_R^{L_i}} = \prod_{i=1}^n \mathcal{F}S(S, R_i) \).

We note that any FuTS \( S = (S, (\theta_i)_{i=1}^n) \) over label sets \( L_i \) and semirings \( R_i \), for \( i = 1 \ldots n \), is a \( \mathcal{V}_R^L \)-coalgebra. In the sequel, we shall use \( \mathcal{V} \) as an abbreviation for \( \mathcal{V}_R^L \) whenever \( L = (L_1, \ldots, L_n) \) and \( R = (R_1, \ldots, R_n) \) are clear from the context. Similarly, and for the sake of readability, we shall often abbreviate \( \mathcal{U}_{\mathcal{V}_R^{L_i}} \) by \( \mathcal{U}_i \).

Under conditions that are generally met, coalgebras come equipped with a natural notion of behavioural equivalence that can act as a reference for strong equivalences, in particular of bisimulation for FuTSs. Below, see Theorem 7, we prove that S-bisimilarity as given by Definition 2, coincides with behavioural equivalence for the functor \( \mathcal{V} \) as given by Definition 3, providing justification for the notion of equivalence defined on FuTSs.

For the notion of behavioural equivalence for the functor \( \mathcal{V} \) to be defined, we need to verify that it possesses a final coalgebra.

**Theorem 6** The functor \( \mathcal{V} \) has a final coalgebra.

**Proof** From the proof of Lemma 5, we obtain that, for each factor \( \mathcal{U}_i \) of \( \mathcal{V} \), there exist sets \( A_i \) and \( B_i \) and a surjective natural transformation \( \eta_i : A_i \times (\cdot)^{B_i} \Rightarrow \mathcal{U}_i, i = 1 \ldots n \). By Lemma 2, it follows that we can find sets \( A \) and \( B \) and a surjective natural transformation \( \eta : A \times (\cdot)^B \Rightarrow \mathcal{V} \). Hence, by Theorem 3, it follows that the functor \( \mathcal{V} \) possesses a final coalgebra \( (\Omega, \omega) \).
Since the functor $\mathcal{V}$ has a final coalgebra, we can speak of the behavioural equivalence $\approx_{\mathcal{V}}$ on any $\mathcal{V}$-coalgebra or, equivalently, of the FuTS $\mathcal{S}$. Writing $\llbracket \cdot \rrbracket_{\mathcal{V}}$ for the final morphism of a $\mathcal{V}$-coalgebra $\mathcal{S}$ into $(\Omega, \omega)$, we have

$$\llbracket \cdot \rrbracket_{\mathcal{V}} = \llbracket \cdot \rrbracket_{\mathcal{U}_1} \times \cdots \times \llbracket \cdot \rrbracket_{\mathcal{U}_n}$$

Next we establish, for a given FuTS $\mathcal{S}$ over $L_1, \ldots, L_n$ and $R_1, \ldots, R_n$, the correspondence of $\mathcal{S}$-bisimulation $\approx_\mathcal{S}$ as given by Definition 4 and the behavioural equivalence $\approx_{\mathcal{V}}$.

**Theorem 7** Let $\mathcal{S} = (S, (\theta_i)_{i=1}^n)$ be a FuTS over the label sets $L_i$ and semirings $R_i$, $i = 1 \ldots n$, and $\mathcal{V}$ as in Definition 4. Then $s_1 \approx_\mathcal{S} s_2 \iff s_1 \approx_{\mathcal{V}} s_2$, for all $s_1, s_2 \in S$.

**Proof** Let $s_1, s_2 \in S$. We first prove $s_1 \approx_\mathcal{S} s_2 \Rightarrow s_1 \approx_{\mathcal{V}} s_2$. So, assume $s_1 \approx_\mathcal{S} s_2$. Let $R \subseteq S \times S$ be an $\mathcal{S}$-bisimulation with $R(s_1, s_2)$. Put $\theta = \theta_1 \times \cdots \times \theta_n$. Note $(S, \theta)$ is a $\mathcal{V}$-coalgebra. We turn the collection of equivalence classes $S/R$ into a $\mathcal{V}$-coalgebra $\mathcal{S}_R = (S/R, \rho_R)$ by putting $\rho_R = \rho_1 \times \cdots \times \rho_n$ where

$$\rho_i([s]_R)(([t]_R) = \sum_{r \in [t]_R} \theta_i(s)(r))$$

for $s, t \in S$, $\ell \in L_i$, $i = 1 \ldots n$. This is well-defined since $R$ is an $\mathcal{S}$-bisimulation: if $R(s, s')$ then we have $\sum_{r \in [t]_R} \theta_i(s)(r) = \sum_{r' \in [s']_R} \theta_i(s')(r')$. The canonical mapping $\varepsilon_R : S \to S/R$ is a $\mathcal{V}$-homomorphism: For $i = 1 \ldots n$, $\ell \in L_i$ and $t \in S$, we have

$$\mathcal{U}_i(\varepsilon_R)(\theta_i(s)(\ell)([t]_R)) = \sum_{r \in [t]_R} \theta_i(s)(r) \quad \text{by definition of } \mathcal{U}_i$$

$$= \sum_{r' \in [s']_R} \theta_i(s)(r') \quad \text{by definition of } \varepsilon_R$$

$$= \rho_i([s]_R)(([t]_R) \quad \text{by definition of } \rho_i$$

Thus, $\mathcal{U}_i(\varepsilon_R) \circ \theta_i = \rho_i \circ \varepsilon_R$. Since $\mathcal{V}(\varepsilon_R) = \prod_{i=1}^n \mathcal{U}_i(\varepsilon_R)$ it follows that $\varepsilon_R$ is a $\mathcal{V}$-homomorphism. Therefore, by uniqueness of a final morphism, we have $\llbracket \cdot \rrbracket_{\mathcal{V}} = \llbracket \cdot \rrbracket_{\mathcal{S}_R} \circ \varepsilon_R$. In particular, with respect to $\mathcal{S}$, this implies $\llbracket s_1 \rrbracket_{\mathcal{V}} = \llbracket s_2 \rrbracket_{\mathcal{V}}$ since $\varepsilon_R(s_1) = \varepsilon_R(s_2)$. Thus, $s_1 \approx_\mathcal{S} s_2$.

For the reverse, $s_1 \approx_\mathcal{S} s_2 \Rightarrow s_1 \approx_{\mathcal{V}} s_2$, assume $s_1 \approx_\mathcal{S} s_2$, i.e. $\llbracket s_1 \rrbracket_{\mathcal{V}} = \llbracket s_2 \rrbracket_{\mathcal{V}}$. Since the map $\llbracket \cdot \rrbracket_{\mathcal{V}} : (S, \theta) \to (\Omega, \omega)$ is a $\mathcal{V}$-homomorphism, the equivalence relation $R_S$ with $R_S(s', s'') \iff \llbracket s' \rrbracket_{\mathcal{V}} = \llbracket s'' \rrbracket_{\mathcal{V}}$ is an $\mathcal{S}$-bisimulation. Suppose $R_S(s', s'')$, i.e. $s' \approx_{\mathcal{V}} s''$, for some $s', s'' \in S$. Assume $\omega = \omega_1 \times \cdots \times \omega_n$. Pick $1 \leq i \leq n$, $\ell \in L_i$, $t \in S$ and assume $\llbracket t \rrbracket_{\mathcal{V}} = \omega_1 \omega_2 \cdots \omega_n$. Since $\llbracket \cdot \rrbracket_{\mathcal{V}} : (S, \theta) \to (\Omega, \omega)$ is a $\mathcal{V}$-homomorphism we have, for $i = 1 \ldots n$, that $\omega_i \cdot \llbracket \cdot \rrbracket_{\mathcal{V}} = \mathcal{U}_i(\llbracket \cdot \rrbracket_{\mathcal{V}}) \circ \theta_i$. Hence, for $s \in S$, it holds that

$$\omega_i(\llbracket s \rrbracket_{\mathcal{V}})(\ell)(w) = \mathcal{U}_i(\llbracket \cdot \rrbracket_{\mathcal{V}})(\theta_i(s)(\ell))(w) = \sum_{r \in [s]_{\mathcal{V}}(\omega_i)} \theta_i(s)(r)(r')$$

Therefore we have

$$\sum_{r \in [s]_{\mathcal{V}}(\omega_i)} \theta_i(s)(r)(r') = \sum_{r \in [s']_{\mathcal{V}}(\omega_i)} \theta_i(s')(r)(r') \quad \text{by definition of } R_S \text{ and } w$$

$$= \omega_i(\llbracket s' \rrbracket_{\mathcal{V}})(\ell)(w) \quad \text{by equation (2)}$$

$$= \omega_i(\llbracket s'' \rrbracket_{\mathcal{V}})(\ell)(w) \quad s' \approx_{\mathcal{V}} s'' \text{ by assumption}$$

$$= \sum_{r \in [s'']_{\mathcal{V}}(\omega_i)} \theta_i(s'')(r)(r') \quad \text{by equation (2)}$$

$$= \sum_{r \in [t]_R} \theta_i(s')(r) \quad \text{by definition of } R_S \text{ and } w$$
Thus, if \( R_S(s', s'') \) then \( \sum_{r \in [i]} \theta_i(s')(r') = \sum_{r \in [i]} \theta_i(s'')(r') \) for all \( t \in S, i = 1 \ldots n, \ell \in L_i \), and therefore \( R_S \) is an \( S \)-bimulation. Since \( \|s_1\|_V = \|s_2\|_V \), it follows that \( R_S(s_1, s_2) \). Thus \( R_S \) is an \( S \)-bimulation relating \( s_1 \) and \( s_2 \). Conclusion, it holds that \( s_1 \approx_S s_2 \).

We continue with relating \( \text{FuTS} \) bisimilarity via behavioural equivalence to coalgebraic bisimulation. We will show that, for a \( \text{FuTS} S \) over \( L = (L_1, \ldots, L_n) \) and \( R = (R_1, \ldots, R_n) \), when seen as a coalgebra, behavioural equivalence \( \approx_V \) and coalgebraic bisimulation \( \sim_V \) coincide. Thus in view of Theorem 7, we will have that \( \text{FuTS} \) bisimilarity \( \approx_S \) and coalgebraic bisimulation \( \sim_{\mathcal{V}} \) are the same. For this, it suffices by Theorem 4 to verify that the functor \( \mathcal{V} \) preserves weak pullbacks. However, our construction below requires that the semirings involved satisfy two additional requirements: (i) existence of a (right) multiplicative inverse for non-zero elements, i.e. for \( r \in R \setminus \{0\} \) it holds that \( r \cdot r' = 1_R \) for some \( r' \in R \); (ii) the zero-sum property, stating that a sum \( r_1 + r_2 \cdots + r_n = 0_R \) if each \( r_i = 0_R, \ i = 1 \ldots n \). Thus, first, non-degenerate quotients exist, and, second, non-zero elements cannot be canceled out. In the concrete case of \( \mathbb{B} \) and \( \mathbb{R}_{>0} \), that will be used in the sequel, these requirements are clearly fulfilled. For readability, we will use standard notation like \( r_1/r_2 \) rather than \( r_1 \cdot r_2^{-1} \).

We first consider the case of a simple \( \text{FuTS} \) and establish the preservation of weak pullbacks for the functor \( \mathcal{U} \) of Definition 3, moving to the functor \( \mathcal{V} \) of Definition 4 of a general \( \text{FuTS} \) afterward.

![Figure 4: Functor \( \mathcal{U} \) preserves weak pullbacks](image)

**Lemma 8** For a set of labels \( L \) and a semiring \( \mathcal{R} \), with multiplicative inverse and satisfying the zero-sum property, the functor \( \mathcal{U} \) of Definition 3 preserves weak pullbacks.

**Proof** See Figure 4. Let \( f : A \to C \) and \( g : B \to C \) be a cospan. Put \( P = \{(a, b) \in A \times B \mid f(a) = g(b)\} \) and let \( \pi_1 : P \to A \) and \( \pi_2 : P \to B \) be the projections. Then \( P \) is the pullback of \( f \) and \( g \). Clearly, \( \mathcal{U}(f) \circ \mathcal{U}(\pi_1) = \mathcal{U}(g) \circ \mathcal{U}(\pi_2) \) as \( f \circ \pi_1 = g \circ \pi_2 \).

Suppose \( F : Q \to \mathcal{U}(A) \) and \( G : Q \to \mathcal{U}(B) \) are such that \( \mathcal{U}(f) \circ F = \mathcal{U}(g) \circ G \). Then we define \( N : Q \to \mathcal{U}(P) \) by

\[
N(q)(\ell)(a, b) = \begin{cases} 
\frac{F(q)(\ell)(a) \cdot G(q)(\ell)(b)}{\mathcal{U}(g)(G(q)(\ell)(f(a)))} & \text{if } \mathcal{U}(g)(G(q)(\ell)(f(a))) \neq 0 \\
0 & \text{otherwise}
\end{cases}
\]

for \( \ell \in L \) and \( (a, b) \in P \). Note that the above construction is well-defined as \( \mathcal{R} \) is assumed to have quotients. We claim that \( F = \mathcal{U}(\pi_1) \circ N \) and \( G = \mathcal{U}(\pi_2) \circ N \) (3)
Note, for \( \ell \in \mathcal{L} \), \((a, b) \in P\) and \(q \in Q\), we have
\[
\mathcal{U}(g)(G(q))(\ell)(f(a)) = \mathcal{U}(g)(G(q))(\ell)(g(b)) = \mathcal{U}(f)(F(q))(\ell)(g(b))
\] (4)
since \(f(a) = g(b)\) and \(\mathcal{U}(f) \circ F = \mathcal{U}(g) \circ G\). So, the definition of \(N\) is symmetric in \(a\) and \(b\).

We verify that \(F = \mathcal{U}(\pi_1) \circ N\). Pick \(q \in Q\), \(\ell \in \mathcal{L}\) and \(a \in A\). We must show \(F(q)(\ell)(a) = \mathcal{U}(\pi_1)(N(q))(\ell)(a)\). We distinguish two cases.

Case I: \(\mathcal{U}(g)(G(q))(\ell)(f(a)) = 0\). Then we have
\[
\mathcal{U}(\pi_1)(N(q))(\ell)(a) = \sum \{ N(q)(\ell)(a, b) \mid b \in B : f(a) = g(b) \} = 0
\]
by definition of \(N\).

On the other hand, if we have \(\mathcal{U}(g)(G(q))(\ell)(f(a)) \neq 0\), then also have \(\mathcal{U}(f)(F(q))(\ell)(f(a)) = 0\) since \(\mathcal{U}(f) \circ F = \mathcal{U}(g) \circ G\). Since
\[
\mathcal{U}(f)(F(q))(\ell)(f(a)) = \sum_{a' \in f^{-1}(f(a))} F(q)(\ell)(a') = \sum_{a' : f(a') = f(a)} F(q)(\ell)(a')
\]
we obtain \(\sum \{ F(q)(\ell)(a') \mid a \in A : f(a') = f(a) \} = 0\). In particular, by the zero-sum property of \(\mathcal{R}\), \(F(q)(\ell)(a) = 0\). Hence, both \(F(q)(\ell)(a) = 0\) and \(\mathcal{U}(\pi_1)(N(q))(\ell)(a) = 0\) and it follows that \(F(q)(\ell)(a) = \mathcal{U}(\pi_1)(N(q))(\ell)(a)\).

Case II: \(\mathcal{U}(g)(G(q))(\ell)(f(a)) \neq 0\). Then we have, since \((a, b) \in \pi_1^{-1}(a)\) iff \(f(a) = g(b)\),
\[
\begin{align*}
\mathcal{U}(\pi_1)(N(q))(\ell)(a) & = \sum_{b : f(a) = g(b)} N(q)(\ell)(a, b) \quad \text{by definition of } \mathcal{U}(\pi_1) \\
& = \sum_{b : f(a) = g(b)} \frac{F(q)(\ell)(a) \cdot G(q)(\ell)(b)}{\mathcal{U}(g)(G(q))(\ell)(f(a))} \quad \text{by definition of } N \\
& = \frac{F(q)(\ell)(a) \cdot \sum_{b : f(a) = g(b)} G(q)(\ell)(b)}{\mathcal{U}(g)(G(q))(\ell)(f(a))} \quad \text{distributivity of } \mathcal{R} \\
& = \frac{F(q)(\ell)(a) \cdot \mathcal{U}(g)(G(q))(\ell)(f(a))}{\mathcal{U}(g)(G(q))(\ell)(f(a))} \quad \text{by definition of } \mathcal{U}(g) \\
& = F(q)(\ell)(a)
\end{align*}
\]

Hence, also for this case, \(F(q)(\ell)(a) = \mathcal{U}(\pi_1)(N(q))(\ell)(a)\). We conclude that \(F = \mathcal{U}(\pi_1) \circ N\).

For proving the claim [3], it remains to check that \(G = \mathcal{U}(\pi_2) \circ N\). Pick \(q \in Q\), \(\ell \in \mathcal{L}\) and \(b \in B\). We now distinguish three cases.

Case I: \(g(b) \notin f[A]\). Then, \(\mathcal{U}(\pi_2)(N(q))(\ell)(b) = \sum_{a : f(a) = g(b)} N(q)(\ell)(a, b) = 0\) as the index set is empty. On the other hand,
\[
\begin{align*}
\sum_{b' : g(b') = g(b)} G(q)(\ell)(b') & = \sum_{b' \in g^{-1}(g(b))} G(q)(\ell)(b') \quad g(b') = g(b) \text{ iff } b' \in g^{-1}(g(b)) \\
& = \mathcal{U}(g)(G(q))(\ell)(g(b)) \quad \text{by definition of } \mathcal{U} \\
& = \mathcal{U}(f)(F(q))(\ell)(g(b)) \quad \text{since } \mathcal{U}(f) \circ F = \mathcal{U}(g) \circ G \\
& = \sum_{a' \in f^{-1}(g(b))} F(q)(\ell)(a) \quad \text{by definition of } \mathcal{U} \\
& = \sum_{a' : f(a) = g(b)} F(q)(\ell)(a) \quad a' \in f^{-1}(g(b)) \text{ iff } f(a) = g(b) \\
& = 0 \quad \text{again because the index set is empty}
\end{align*}
\]
Since \( \sum_{b' \geq g(b')} G(q)(\ell)(b') = 0 \), it follows from the zero-sum property of \( R \) that \( G(q)(\ell)(b') = 0 \) for each \( b' \in B \) such that \( g(b') = g(b) \). In particular, \( G(q)(\ell)(b) = 0 \) and therefore \( G(q)(\ell)(b) = U(\pi_2)(N(q)))(\ell)(b) \).

Case II: \( g(b) = f(a) \) for some \( a \in A \) and \( U(g)(G(q)(\ell)(f(a)) = 0 \). Note that the latter equality is independent of the choice of \( a \) by equation (4). On the one hand, we have \( U(\pi_2)(N(q))(\ell)(b) = \sum_{a \in f(a) = g(b)} N(q)(\ell)(a,b) = 0 \), since by definition of \( N \) we have \( N(q)(\ell)(a,b) = 0 \) for each \( a \in A \) such that \( f(a) = g(b) \) in this case, as \( U(g)(G(q)(\ell)(f(a)) = 0 \). On the other hand, \( \sum_{b' \geq g(b')} G(q)(\ell)(b') = U(g)(G(q)(\ell)(g(b))) = U(g)(G(q)(\ell)(f(a)) = 0 \) by assumption. Again, by the zero-sum property of \( R \), we obtain in particular that \( G(q)(\ell)(b) = 0 \). It follows that \( G(q)(\ell)(b) = U(\pi_2)(N(q))(\ell)(b) \).

Case III: \( g(b) = f(a) \) for some \( a \in A \) and \( U(g)(G(q)(\ell)(f(a)) \neq 0 \). Then it holds that

\[
U(\pi_2)(N(q))(\ell)(b) = \sum_{a \in \pi_2^{-1}(b)} N(q)(\ell)(a,b) = \sum_{a : f(a) = g(b)} N(q)(\ell)(a,b) = \sum_{a : f^{-1}(g(b))} N(q)(\ell)(a,b) = \sum_{a : f^{-1}(g(b))} F(q)(\ell)(a) = G(q)(\ell)(b)
\]

Thus, also in this case \( G(q)(\ell)(b) = U(\pi_2)(N(q))(\ell)(b) \) and we conclude that \( G = U(\pi_2) \circ N \). This proves the claim.

Now, let \( W \) with span \( w_1 : W \to A \) and \( w_2 : W \to B \) be any weak pullback for \( f \) and \( g \). See again Figure 1. Since \( W \) is a weak pullback of \( f \) and \( g \), and \( \pi_1 \) and \( \pi_2 \) satisfy \( f \circ \pi_1 = g \circ \pi_2 \), there exists a mapping \( m : P \to W \) such that \( \pi_1 = w_1 \circ m \) and \( \pi_2 = w_2 \circ m \). Suppose again that \( F : Q \to A \) and \( G : Q \to B \) is a span such that \( U(f) \circ F = U(g) \circ G \). To prove the lemma we need to show that there exists \( M : Q \to W \) such that \( F = U(w_1) \circ M \) and \( G = U(w_2) \circ M \). Put \( M = U(m) \circ N \). Clearly, \( U(\pi_1) = U(w_1) \circ U(m) \) and \( U(\pi_2) = U(w_2) \circ U(m) \). Therefore we have, by equation (5), \( F = U(\pi_1) \circ N = U(w_1) \circ U(m) \circ N = U(w_1) \circ M \), and similarly \( G = U(w_2) \circ M \), as was to be shown. In [Bonchi et al. (2011)], Section 2.2], in the setting of weighted automata, a counter example is given for the general case of Lemma [8]. The construction there involves the semiring \( \mathbb{Z} \) and, specifically the fact that \( 1 + (-1) = 0 \). Thus the construction of [Bonchi et al. (2011)] exploits in particular that the semiring involved does not have the zero-sum property.

The proof of Lemma [8] in line with [de Vink & Rutten (1999)], explicitly constructs the mediating morphism \( N \). One can also appeal to a generalization of the so-called Row-Column Theorem, as coined by Moss in [Moss (1999)]. An outline of such a proof is sketched in [Klin (2009)]. For a version of the Row/Column Theorem dealing with infinite matrices, see [Sokolova et al. (2005)] Lemma 3.5.5. In a setting with finite sums, [Gumm & Schröder (2001)] provides a characterization of the monoids \( M \) for which the functor \( \mathcal{FS}(\cdot, M) \) preserves weak pullbacks, viz. those monoids with have the zero-sum property and for which the Row/Column Theorem holds too.

With the Lemma [8] in place we are in a position to relate behavioural equivalence and coalgebraic bisimulation as induced by a FuTS. Note, as for Lemma [8] the proof relies on the semirings to have the zero-sum property.
Theorem 9 Let $S = (S, (\theta_i)_{i=1}^n)$ be a FuTS over the label sets $L_i$ and semirings $R_i$, $i = 1 \ldots n$, with multiplicative inverse and satisfying the zero-sum property. For $\mathcal{V}$ as in Definition 4 it holds that $\approx_{\mathcal{V}}$ and $\approx_{\mathcal{V}}'$ coincide.

Proof We recall that the functor $\mathcal{V}$ on Set is defined by $\mathcal{V} = \prod_{i=1}^n \mathcal{U}_i$, where, for $i = 1 \ldots n$, $\mathcal{U}_i = TFS(\cdot, R_i)^L_i$. By Lemma 8 each factor $\mathcal{U}_i$, $i = 1 \ldots n$, preserves weak pullbacks.

Let $f : A \to C$ and $g : B \to C$ be a cospan and let $W$ with $w_1 : W \to A$ and $w_2 : W \to B$ be a weak pullback for $f$ and $g$. Suppose $F : Q \to \mathcal{V}(A)$ and $G : Q \to \mathcal{V}(B)$ is a span such that $\mathcal{V}(f) \circ F = \mathcal{V}(g) \circ G$. See Figure 5.

By Lemma 8 we have, for all $i = 1 \ldots n$, that $\mathcal{U}_i(W)$ with $\mathcal{U}_i(w_1)$ and $\mathcal{U}_i(w_2)$ is a weak pullback of $\mathcal{U}_i(f)$ and $\mathcal{U}_i(g)$. Since, for the projections $\pi^A_i : \mathcal{V}(A) \to \mathcal{U}_i(A)$, $\pi^B_i : \mathcal{V}(B) \to \mathcal{U}_i(B)$ and $\pi^C_i : \mathcal{V}(C) \to \mathcal{U}_i(C)$, we have $\mathcal{U}_i(f) \circ \pi^A_i \circ F = \pi^F_i \circ \mathcal{V}(f) \circ F = \pi^C_i \circ \mathcal{V}(g) \circ G = \mathcal{U}_i(g) \circ \pi^B_i \circ G$, there exists a mapping $n_i : Q \to \mathcal{U}_i(W)$ such that $n_i \cdot F = \mathcal{U}_i(w_1) \circ n_i$ and $n_i \cdot G = \mathcal{U}_i(w_2) \circ n_i$ by the weak pullback property, for each index $i = 1 \ldots n$. Define $N : Q \to \mathcal{V}(W)$ by $N = n_1 \times \cdots \times n_n$. Then we have, for any $q \in Q$,

\[(\mathcal{V}(w_1) \circ N)(q) = \mathcal{V}(w_1)(\langle n_1(q), \ldots, n_n(q) \rangle) = \langle \mathcal{U}_1(w_1)(n_1(q)), \ldots, \mathcal{U}_n(w_1)(n_n(q)) \rangle = \langle \pi^A_1(F(q)), \ldots, \pi^A_n(F(q)) \rangle = F(q)\]

Thus $\mathcal{V}(w_1) \circ N = F$. Similarly, $\mathcal{V}(w_2) \circ N = G$. We conclude that $\mathcal{V}(W)$ with $\mathcal{V}(w_1)$ and $\mathcal{V}(w_2)$ is a weak pullback of $\mathcal{V}(f)$ and $\mathcal{V}(g)$ and that the functor $\mathcal{V}$ preserves weak pullbacks. Therefore, by Theorem 4 the result follows.

![Figure 5: Functor $\mathcal{V}$ preserves weak pullbacks too, since each factor $\mathcal{U}_i$ does.](image)

In the sequel we will provide a FuTS semantics for three representative process languages, one with qualitative non-determinacy, one quantitative with stochastic-time non-determinacy, and a mixed one having both qualitative and quantitative stochastic-time non-determinacy. For these languages we will establish that their standard notions of strong equivalence as known in the literature coincide with the notion of strong bisimulation as induced by the FuTS semantics. The results of this section imply that the standard notions of strong equivalence on the one hand, and behavioural equivalence and coalgebraic bisimulation on the other hand, are all the same. The notion of bisimulation for FuTS plays an intermediary role, it bridges between the standard notion of concrete equivalence and the abstraction notions from coalgebra.

5 FuTS-semantics for a elementary process language

As a first example of a formal semantics based on FuTSs and of an illustration of the comparison with standard semantics using LTSs, we first consider a minimal process language common
to many process algebras [Hoare(1985), Milner(1989), Baeten et al.(2009)] involving action prefix, non-deterministic choice and recursion via process variables. Then we consider an extension of the language with a parallel operator. The language comprises the basis for many of the SPCs proposed in the literature, and in particular those we discuss in Sections 6 and 7. In fact, the language is a sublanguage of IML (see Section 7).

**Definition 5** Let $\mathcal{A}$ be a set of actions, ranged over by $a$, and let $X$ be the set of constants, or process variables, ranged over by $X$. The set $\mathcal{P}_{elm}$ of ‘elementary’ process terms is given by the grammar $P ::= \text{nil} | a.P | P + P | X$.

We associate with each $X \in X$ a unique process $P \in \mathcal{P}_{elm}$, notation $X := P$. It is required that each occurrence of a constant in the body $P$ of the constant definition is in the scope of a prefix.

The semantics of elementary processes is given as a FuTS over the action set $\mathcal{A}$ and the semiring of booleans $\mathbb{B}$.

**Definition 6** The FuTS semantics of $\mathcal{P}_{elm}$ is given by the simple FuTS $S_{elm} = (\mathcal{P}_{elm}, \rightarrow_{elm})$ where the transition relation $\rightarrow_{elm} \subseteq \mathcal{P}_{elm} \times \mathcal{A} \times \mathcal{TFS(\mathcal{P}_{elm}, \mathbb{B})}$ is the least relation satisfying the rules of Figure 6.

![Figure 6: FuTS semantics for elementary processes.](image)

The \text{nil} process does not display any activity. Therefore, for every action $a \in \mathcal{A}$, the function that records the possible continuations of \text{nil} has an empty support, i.e. $\text{nil} \rightarrow_{elm} [1]\mathbb{B}$, which is exactly rule (NIL). Recall, $\text{false}$ is the 0-element of the semiring $\mathbb{B}$ and $[1]: \mathcal{P}_{elm} \rightarrow \mathbb{B}$ is defined by $[1](P) = \text{false}$ for every $P \in \mathcal{P}_{elm}$. An action-prefix process $a.P$ executes $a$ and continues to behave as $P$. This is captured by rule (PREF1). For any other action $b$, there is no continuation of $a.P$, see rule (PREF2). With respect to the action $b$ the processes $a.P$ and $\text{nil}$ behave the same.

In dealing with non-deterministic choice we take advantage of the additive structure of the semiring, that extends pointwise to functions. If $P_1, \ldots, P_n$ are all possible continuations for the process $P$ after executing $a$, and $Q_1, \ldots, Q_m$ are all possible continuations for the process $Q$ after executing $a$, then after doing $a$ the process $P + Q$ has the possibilities $P_1, \ldots, P_n$ as well as $Q_1, \ldots, Q_m$. In case $\mathcal{P} = [P_i \mapsto \text{true}]_{i=1}^n$ and $\mathcal{Q} = [Q_j \mapsto \text{true}]_{j=1}^m$, this is represented in rule (CHO) by the function $\mathcal{P} + \mathcal{Q} = [P_i \mapsto \text{true}]_{i=1}^n + [Q_j \mapsto \text{true}]_{j=1}^m = [P \mapsto \text{true}, \ldots, P_n \mapsto \text{true}, \ldots, Q_1 \mapsto \text{true}, \ldots, Q_m \mapsto \text{true}].$ The rule (CNS) for the constant simply copies the transition for the ‘body’ $P$ of the constant $X$ if we have $X := P$. Note, that such is well-defined as $P$ is required to be guarded.

It can be straightforwardly shown by guarded induction that the FuTS $S_{elm}$ is total and deterministic: Clearly, for each $a \in \mathcal{A}$ there exists a unique $\mathcal{P} \in \mathcal{TFS(\mathcal{P}_{elm}, \mathbb{B})}$ with $\text{nil} \rightarrow_{elm} \mathcal{P}$, viz. $[1]$. Also, for each $b \in \mathcal{A}$, either $b = a$ or $b \neq a$, there exists a unique $\mathcal{P} \in \mathcal{TFS(\mathcal{P}_{elm}, \mathbb{B})}$ with $a.P \rightarrow_{elm} \mathcal{P}$. In case $a = b$ we have $\mathcal{P} = [P \mapsto \text{true}]$, in case $a \neq b$ we have $\mathcal{P} = [1]$. Assuming there exist, for given $a \in \mathcal{A}$, unique $\mathcal{P}$ and $\mathcal{Q}$ such that $P \rightarrow_{elm} \mathcal{P}$ and $Q \rightarrow_{elm} \mathcal{Q}$, respectively, it follows, since
only rule (CHO) applies, that there exists a unique \( \mathcal{P} \in \mathcal{FS}(\mathcal{P}_{\text{elm}}, \mathbb{B}) \), namely \( \mathcal{P} + \mathcal{D} \) that aggregates the results for \( P \), given by \( \mathcal{P} \), and the results for \( Q \), given by \( \mathcal{D} \). Finally, for \( X := P \), by the induction hypothesis there exists a unique \( \mathcal{P} \) such that \( P \xrightarrow{a} \mathcal{P} \), for \( a \in \mathcal{A} \). Then, by virtue of rule (CNS), \( \mathcal{P} \) is also unique such that \( X \xrightarrow{a} \mathcal{P} \).

In the following we will use \( \theta_{\text{elm}} : \mathcal{P}_{\text{elm}} \rightarrow \mathcal{A} \rightarrow \mathcal{FS}(\mathcal{P}_{\text{elm}}, \mathbb{B}) \) for the function corresponding to the relation \( \rightarrow_{\text{elm}} \). We have, \( P \xrightarrow{a} \mathcal{P} \) iff \( \theta_{\text{elm}}(P)(a) = \mathcal{P} \).

In Figure 7 we provide the SOS for elementary processes in the LTS-based approach defining the transition relation \( \rightarrow_{\text{elm}} \). We have \( \rightarrow_{\text{elm}} \subseteq \mathcal{P}_{\text{elm}} \times \mathcal{A} \times \mathcal{P}_{\text{elm}} \). We discuss the various differences of the FuTS and LTS semantics. In the standard semantics there is no rule for the \( \text{nil} \) process, i.e. there is no transition for \( \text{nil} \). The FuTS semantics provides \( \text{nil} \xrightarrow{a}_{\text{elm}} \) for every action \( a \). However, the latter expresses \( \theta_{\text{elm}}(\text{nil})(a)(P') = 0 \) for every \( a \in \mathcal{A} \) and \( P' \in \mathcal{P}_{\text{elm}} \), or, in standard terminology, \( \text{nil} \) has no transition. The standard approach provides only one rule for the prefix construct, rule (PREF). The FuTS approach has two rules for this, rule (PREF1) and (PREF2), as it also explicitly expresses by way of rule (PREF2) that there is no standard transition labeled with another label than \( a \) for the process \( a.P \).

The choice rules (CHOa) and (CHOb) for the LTS semantics similarly differ from the choice rule (CHO) for the FuTS semantics. In the standard approach the non-determinism is resolved by choosing the rule. The FuTS semantics, developed with stochastic process languages in mind where multiplicities matter (see Sections 6 and 7), combines the two branches into one function. The treatment of process variables is the same for the two approaches.

\[
\begin{align*}
\text{(PREF)} & \quad a.P \xrightarrow{a} P & \quad \text{(CNS)} & \quad \frac{P \xrightarrow{a} P'}{X \xrightarrow{a} P'} \\
\text{(CHOa)} & \quad \frac{P \xrightarrow{a} P'}{P + Q \xrightarrow{a} P'} & \quad \text{(CHOb)} & \quad \frac{Q \xrightarrow{a} Q'}{P + Q \xrightarrow{a} Q'}
\end{align*}
\]

Figure 7: Standard SOS for elementary processes.

As we will show, see Theorem 11 for \( S_{\text{elm}} = (\mathcal{P}_{\text{elm}}, \rightarrow_{\text{elm}}) \) like for other FuTSs to follow, the standard notion of strong bisimulation \([\text{Park}(1981), \text{Milner}(1980)]\) identifies the same processes as \( S_{\text{elm}} \)-bisimulation as obtained from Definition 2 and denoted by \( \equiv_{S_{\text{elm}}} \). We first observe the following lemma.

**Lemma 10** Let \( P \in \mathcal{P}_{\text{elm}} \) and \( a \in \mathcal{A} \). Suppose \( P \xrightarrow{a} \mathcal{P} \). Then \( \mathcal{P}(R) \iff P \xrightarrow{a} R \), for all \( R \in \mathcal{P}_{\text{elm}} \).

**Proof** Guarded induction on \( P \): Clear for \( \text{nil} \), as \( \mathcal{P}(R) = [\mathcal{P} \equiv(\mathcal{A}) = \text{false} \) and \( \text{nil} \xrightarrow{a} \mathcal{P} \) for no \( R \in \mathcal{P}_{\text{elm}} \). For \( b.P \), we have \( \mathcal{P}(R) \iff b = a \) and \( [P \iff \text{true}](R) \iff b = a \) and \( P = R \), and also \( b.P \xrightarrow{a} \mathcal{P} \iff b = a \) and \( P = R \). For \( P + Q \), it holds that \( \mathcal{P} = \mathcal{P}' + \mathcal{P}'' \) where \( P \xrightarrow{a} \mathcal{P} \) and \( Q \xrightarrow{a} \mathcal{P}'' \). Thus \( \mathcal{P}(R) \iff (\mathcal{P}' \oplus \mathcal{P}'')(R) \iff \mathcal{P}'(R) \) or \( \mathcal{P}''(R) \) iff, by induction hypothesis, \( P \xrightarrow{a} \mathcal{P} \) or \( Q \xrightarrow{a} \mathcal{P} \) if \( P + Q \xrightarrow{a} \mathcal{P} \). For \( X \) with \( X := P \), we have \( X \xrightarrow{a} \mathcal{P} \iff P \xrightarrow{a} \mathcal{P} \), and \( X \xrightarrow{a} \mathcal{P} \iff P \xrightarrow{a} \mathcal{P} \). Thus \( \mathcal{P}(R) \iff, \) by induction hypothesis, \( P \xrightarrow{a} \mathcal{P} \) if \( X \xrightarrow{a} \mathcal{P} \).

Now, following the usual terminology, a relation \( R \subseteq \mathcal{P}_{\text{elm}} \times \mathcal{P}_{\text{elm}} \) is called a strong bisimulation on \( \mathcal{P}_{\text{elm}} \) if \( P_1 \xrightarrow{a} \mathcal{P}_1 \Rightarrow \exists \mathcal{P}'_2 : P_2 \xrightarrow{a} \mathcal{P}_2 \wedge R(P_1, P_2') \) and \( P_2 \xrightarrow{a} \mathcal{P}_2 \Rightarrow \exists \mathcal{P}'_1 : P_1 \xrightarrow{a} \mathcal{P}_1 \).
Thus, the first transfer condition is met. By symmetry it follows that $R$ is a strong bisimulation on $\mathcal{P}_{elm}$ and $R(P_1, P_2)$. Thus $P_1 \sim_{elm} P_2$.

Next we augment the collection of elementary processes with a(n extension of a) CSP-like parallel operator. Its treatment is slightly involved since a little extra semantic machinery for continuation functions to reflect the syntactic operator is needed. As this obscures a bit the general explanation, we kept initially the parallel construct out of Definition[5] as our main purpose of considering a qualitative process language here is to illustrate the overall approach with FuTS in the quantitative setting.

\[ P'_1 \land R(P'_1, P'_2) \] whenever $R(P_1, P_2)$. Two elements $P_1, P_2 \in \mathcal{P}_{elm}$ are called strongly bisimilar if there exists a strong bisimulation $R$ on $\mathcal{P}_{elm}$ relating $P_1$ and $P_2$, notation $P_1 \sim_{elm} P_2$. It is well-known that we can require $R$ to be an equivalence relation, a so-called strong bisimulation equivalence, as the equivalence closure of $R$ is a strong bisimulation if $R$ is.

In order to relate strong bisimilarity and FuTS-bisimilarity we observe the following: If $R$ is a strong bisimulation equivalence, with equivalence classes $\{P\}_R$ for $P \in \mathcal{P}_{elm}$, then it holds for $P_1, P_2 \in X$ with $R(P_1, P_2)$ that

\[ \exists P' \in \{P\}_R : P_1 \xrightarrow{a_{elm}} P' \iff \exists P'' \in \{P\}_R : P_2 \xrightarrow{a_{elm}} P'' \] (5)

For, if $P_1 \xrightarrow{a_{elm}} P'$ and $R(P', P)$, then, since $R$ is a bisimulation, for some suitable $P''$ we have $P_2 \xrightarrow{a_{elm}} P''$, $R(P'', P')$ and $R(P', P)$, but then also $P_2 \xrightarrow{a_{elm}} P''$ and $R(P'', P)$, since $R$ is an equivalence. We use this in the proof that strong bisimilarity $\sim_{elm}$ and FuTS-bisimilarity $\approx_{S_{elm}}$ for $\mathcal{S}_{elm}$ coincide.

**Theorem 11** For any two processes $P_1, P_2 \in \mathcal{P}_{elm}$, it holds that $P_1 \sim_{elm} P_2$ iff $P_1 \approx_{S_{elm}} P_2$.

**Proof** Pick $P_1, P_2 \in \mathcal{P}_{elm}$. To verify that $P_1 \sim_{elm} P_2$ implies $P_1 \approx_{S_{elm}} P_2$ we assume there exists an equivalence relation $R$ that is a strong bisimulation such that $R(P_1, P_2)$. Then, for arbitrary $a \in \mathcal{A}$, we have

\[
\sum_{Q' \in \{Q\}_R} \theta_{elm}(P_1)(a)(Q') \iff \exists Q' \in \{Q\}_R : P_1 \xrightarrow{a_{elm}} Q' \quad \text{by Lemma[10]}
\]

\[
\iff \exists Q' \in \{Q\}_R : P_1 \xrightarrow{a_{elm}} Q' \quad \text{by equation (5)}
\]

\[
\iff \exists Q'' \in \{Q\}_R : P_2 \xrightarrow{a_{elm}} Q'' \quad \text{by Lemma[10]}
\]

\[
\iff \exists Q' \in \{Q\}_R : P_2 \xrightarrow{a_{elm}} Q' \quad \text{by Lemma[10]}
\]

\[
\iff \sum_{Q' \in \{Q\}_R} \theta_{elm}(P_2)(a)(Q')
\]

for all $Q \in \mathcal{P}_{elm}$, i.e. for all equivalence classes of $R$. Hence, $R$ is also an $S_{elm}$-bisimulation and, since $R(P_1, P_2)$, we conclude $P_1 \approx_{S_{elm}} P_2$.

To show the reverse, that $P_1 \approx_{S_{elm}} P_2$ implies $P_1 \sim_{elm} P_2$, let $R$ be an $S_{elm}$-bisimulation with $R(P_1, P_2)$. We have

\[
P_1 \xrightarrow{a_{elm}} Q' \quad \iff \quad \theta_{elm}(P_1)(a)(Q') \quad \text{by Lemma[10]}
\]

\[
\iff \sum_{Q' \in \{Q\}_R} \theta_{elm}(P_1)(a)(Q) \quad \text{property $B$}
\]

\[
\iff \sum_{Q' \in \{Q\}_R} \theta_{elm}(P_1)(a)(Q) \quad \text{property $B$}
\]

\[
\iff \exists Q' \in \{Q\}_R : \theta_{elm}(P_2)(a)(Q) \quad 	ext{R is an $S_{elm}$-bisimulation}
\]

\[
\iff \exists Q'' \in \{Q\}_R : \theta_{elm}(P_2)(a)(Q') \quad \text{property $B$}
\]

\[
\iff \exists Q'' : P_2 \xrightarrow{a_{elm}} Q'' \iff R(Q', Q'') \quad \text{by Lemma[10]}
\]

Thus, the first transfer condition is met. By symmetry it follows that $R$ is a strong bisimulation on $\mathcal{P}_{elm}$ and $R(P_1, P_2)$. Thus $P_1 \sim_{elm} P_2$.\[\]

ASCENS 20
In other words, if definitions, we get R
Note, if existent, a split-up = R
functions directly, cf. [Cardelli & Mardare(2010)]. For the carrier P
for state-to-function relation ⋮ P
is, in the latter case a
specifically, for any subset P
The computational intuition is that in ASCENS 21
the former case Q remains as, in the latter case P. Following the description introduced in Section 2 for injective binary operation

\[ \parallel A \parallel \]

as the semantical counterpart, by putting \( (\mathcal{P}_2 \parallel A \mathcal{P}_2)(R) = \mathcal{P}_1(R_1) \land \mathcal{P}_2(R_2) \) if \( R = R_1 \parallel A R_2 \). More specifically, for any subset \( A \subseteq \mathcal{A} \),

\[
\begin{align*}
(\mathcal{P}_2 \parallel A \mathcal{P}_2)[Q_i & \mapsto \text{true}, \ldots, Q_m \mapsto \text{true} ](R) \\
= \sum_{R_1: R_1 \parallel A R_2} (P_1 \mapsto \text{true}, \ldots, P_n \mapsto \text{true}) & \land (Q_1 \mapsto \text{true}, \ldots, Q_m \mapsto \text{true}) \parallel R_2 \\
= \bigvee_{R_1: R_1 \parallel A R_2} (\bigvee_{j=1}^m (R_1 = P_j)) & \lor (\bigvee_{j=1}^m (R_2 = Q_j)) \\
= \bigvee_{R_1: R_1 \parallel A R_2} (R = P_1) & \parallel Q_j \\
\end{align*}
\]

Note, if existent, a split-up \( R = R_1 \parallel A R_2 \) is unique here, as we do not deal with congruence classes directly, cf. [Cardelli & Mardare(2010)]. For the carrier \( \mathcal{P}_elm \) and the semiring \( \mathbb{B} \) we have the Dirac functions \( D_Q : \mathcal{P}_elm \to \mathbb{B} \) and \( D_Q = [Q \mapsto \text{true}] \), for \( Q \in \mathcal{P}_elm \). Thus, unfolding the various definitions, we get

\[
((\mathcal{P} \parallel A D_Q) + (D_P \parallel A \mathcal{D}))(R)
\]

\[ \iff \] by definition of + on \( \mathcal{F}(\mathcal{P}_elm, \mathbb{B}) \)

\[
(\mathcal{P} \parallel A D_Q)(R) \lor (D_P \parallel A \mathcal{D})(R)
\]

\[ \iff \] by definition of \( \parallel A \parallel \) on \( \mathcal{F}(\mathcal{P}_elm, \mathbb{B}) \)

\[
((R = R_1 \parallel A R_2) \land \mathcal{P}(R_1) \land D_Q(R_2)) \lor ((R = R_1 \parallel A R_2) \land D_P(R_1) \land \mathcal{D}(R_2))
\]

\[ \iff \] \( (R = R_1 \parallel A Q) \land \mathcal{P}(R_1) \lor (R = P \parallel A R_2) \land \mathcal{D}(R_2) \)

In other words, if \( P \overset{a}{\Rightarrow} \mathcal{P} \) and \( Q \overset{a}{\Rightarrow} \mathcal{D} \) then \((\mathcal{P} \parallel A D_Q) + (D_P \parallel A \mathcal{D}))(R) = \text{true} \) if \( R = R_1 \parallel A Q \) and \( \mathcal{P}(R_1) = \text{true} \) or \( R = P \parallel A R_2 \) and \( \mathcal{D}(R_2) = \text{true} \).
Rule (PAR1) deals with interleaving for actions $a \notin \mathcal{A}$ only. Synchronization of $P$ and of $Q$ for $P \parallel_A Q$ on actions $a \in A$ is incorporated in rule (PAR2). As both process $P$ and process $Q$ need to progress, with respect to the action $a \in A$, the continuation $\mathcal{P}$ for $P$ and the continuation $\mathcal{Q}$ for $Q$ can be combined directly. For the extended state-to-function transition relation totality and determinacy is straightforward to check.

**Example** Consider the process $P = a.P_1 + b.P_2 + c.P_3 + d.P_4$ and $Q = a.Q_1 + b.Q_2 + b.Q_3 + c.Q_4$. Let $A = \{a, b\}$. Then we have

\[
P \overset{a}{\rightarrow}_{elm} [P_1 \mapsto true] \quad \text{and} \quad Q \overset{a}{\rightarrow}_{elm} [Q_1 \mapsto true]
\]

\[
P \overset{b}{\rightarrow}_{elm} [P_2 \mapsto true] \quad \text{and} \quad Q \overset{b}{\rightarrow}_{elm} [Q_2 \mapsto true, Q_3 \mapsto true]
\]

\[
P \overset{c}{\rightarrow}_{elm} [P_3 \mapsto true] \quad \text{and} \quad Q \overset{c}{\rightarrow}_{elm} [Q_4 \mapsto true]
\]

\[
P \overset{d}{\rightarrow}_{elm} [P_4 \mapsto true] \quad \text{and} \quad Q \overset{d}{\rightarrow}_{elm} [Q_5 \mapsto true]
\]

Therefore, by rule (PAR2) for actions $a$ and $b$ and by rule (PAR1) for actions $c$ and $d$, we have

\[
P \parallel_A Q \overset{a}{\rightarrow}_{elm} [P_1 \parallel_A Q_1 \mapsto true]
\]

\[
P \parallel_A Q \overset{b}{\rightarrow}_{elm} [P_2 \parallel_A Q_2 \mapsto true, P_2 \parallel_A Q_3 \mapsto true]
\]

\[
P \parallel_A Q \overset{c}{\rightarrow}_{elm} [P_3 \parallel_A Q \mapsto true, P \parallel_A Q_4 \mapsto true]
\]

\[
P \parallel_A Q \overset{d}{\rightarrow}_{elm} [P_4 \parallel_A Q \mapsto true]
\]

The standard SOS rules for the operator $\parallel_A$, for $A \subseteq \mathcal{A}$ are given in Figure 9, the two rules (PAR1a) and (PAR1b) for the interleaving case, the rule (PAR2) for the synchronizing case. For the extended language of elementary processes Lemma 10 holds as well. For the corresponding extension of the proof only one other case for the induction step needs to be considered (split up in a subcase for an action $a \notin A$ and one for $a \in A$):

For the case $P_1 \parallel_A P_2$ where $P_1 \parallel_A P_2 \overset{a}{\rightarrow}_{elm} \mathcal{P}$ for $a \notin A$, suppose $P_1 \overset{a}{\rightarrow}_{elm} \mathcal{P}_1$ and $P_2 \overset{a}{\rightarrow}_{elm} \mathcal{P}_2$. We have, for $R \in \mathcal{P}_{elm}$,

\[
\mathcal{P}(R) \iff \text{rule (PAR1) for } \rightarrow_{elm} \\text{ for } \mathcal{P}_1 \parallel_A \mathcal{P}_2 \text{ for } (R) \\text{ for } \mathcal{P}(R)
\]

\[
\implies \text{definition } \parallel_A \text{ and } \mathcal{D}_{P_1}, \mathcal{D}_{P_2}
\]

\[
(\exists R_1 : R = R_1 \parallel_A P_2 \land \mathcal{P}_1(R_1)) \lor (\exists R_2 : R = P_1 \parallel_A R_2 \land \mathcal{P}_2(R_2))
\]

\[
\implies \text{by induction hypothesis}
\]

\[
(\exists R_1 : R = R_1 \parallel_A P_2 \land P_1 \overset{a}{\rightarrow}_{elm} R_1) \lor (\exists R_2 : R = P_1 \parallel_A R_2 \land P_2 \overset{a}{\rightarrow}_{elm} R_2)
\]

\[
\implies \text{both (PAR1) rules for } \rightarrow_{elm}
\]

\[
(\exists R_1 : R = R_1 \parallel_A P_2 \land P_1 \parallel_A P_2 \overset{a}{\rightarrow}_{elm} R_1 \parallel_A P_2) \lor
\]

\[
(\exists R_2 : R = P_1 \parallel_A R_2 \land P_1 \parallel_A P_2 \overset{a}{\rightarrow}_{elm} P_1 \parallel_A R_2)
\]

\[
\implies P_1 \parallel_A P_2 \overset{a}{\rightarrow}_{elm} R
\]

For the case $P_1 \parallel_A P_2$ where $P_1 \parallel_A P_2 \overset{a}{\rightarrow}_{elm} \mathcal{P}$ for $a \in A$, suppose $P_1 \overset{a}{\rightarrow}_{elm} \mathcal{P}_1$ and $P_2 \overset{a}{\rightarrow}_{elm} \mathcal{P}_2$. We have, for $R \in \mathcal{P}_{elm}$,
Figure 9: Standard semantics for the parallel operator $\parallel_A$

\begin{align*}
(\text{PAR1a}) & & P \xrightarrow{a_{\text{elm}}} P' & & a \notin A \\
& & P \parallel_A Q \xrightarrow{a_{\text{elm}}} P' \parallel_A Q & & a \notin A \\
(\text{PAR1b}) & & Q \xrightarrow{a_{\text{elm}}} Q' & & a \notin A \\
& & P \parallel_A Q \xrightarrow{a_{\text{elm}}} P \parallel_A Q' & & a \notin A
\end{align*}

With the extension of Lemma 10 in place it follows that Theorem 11 is also valid for the extended set of processes.

Restriction and relabelling operators can be handled straightforwardly in the FuTS framework. A treatment of a CCS-style parallel operator with FuTS proceeds along the same lines, in particular when the set of actions $A$ is assumed to be finite. For a countably infinite set $A$, the complication arises that potentially unbounded sums need to be considered, as for every matching pair of actions $a$ and $\bar{a}$ such that $P_1 \xrightarrow{a} P_1$ and $P_2 \xrightarrow{\bar{a}} P_2$ the combined continuation of $P_1$ and $P_2$ needs to be taken into account. It can be proven, but we do not do so here, that the summations in the particular synchronization rule for the parallel operator are well-defined, using the fact that the processes involved admit finitely many transition only to a continuation different from $[]_B$.

6 FuTS Semantics of PEPA

Next we will consider a significant part of the process algebra PEPA [Hillston(1996)], including the parallel operator implementing the scheme of so-called minimal apparent rates, and provide a FuTS semantics for it. We point out that there is no technical difficulty in extending the FuTS approach to the full language; we do not do so here since its treatment does not add any conceptual benefit to the present paper. We will show that PEPA’s notion of equivalence $\sim_{pepa}$, called strong equivalence in [Hillston(1996)], fits with the bisimilarity $\simeq_{S_{pepa}}$ arising from the FuTS semantics.

Definition 7 The set $P_{pepa}$ of PEPA processes is given by the grammar below:

\[
P ::= \text{nil} \mid (a, \lambda).P \mid P + P \mid P \parallel P \mid X
\]

where $a$ ranges over the set of actions $A$, $\lambda$ over $\mathbb{R}_{>0}$, $A$ over the set of finite subsets of $A$, and $X$ over the set of constants $X$.\hfill\bull
For $X \in X$, the notation $X := P$ indicates that the process $P$ is associated with the constant $X$. It is required that each occurrence of a process constant in the body $P$ of the definition $X := P$ is guarded by a prefix.

PEPA, like many other SPCs, e.g. [Hermans et al.(1998)], [Bernardo & Gorrieri(1998)], couples actions and rates. The prefix $(a, \lambda)$ of the process $(a, \lambda)P$ expresses that the duration of the execution of the action $a \in \mathcal{A}$ is sampled from a random variable with an exponential distribution of rate $\lambda$. The CSP-like parallel composition $P \parallel Q$ of a process $P$ and a process $Q$ for a set of actions $A \subseteq \mathcal{A}$ allows for the independent, asynchronous execution of actions of $P$ or $Q$ not occurring in the subset $A$, on the one hand, and requires the simultaneous, synchronised execution of $P$ and $Q$ for the actions occurring in $A$, on the other hand. The FuTS-semantics of the fragment of PEPA that we consider here, is given in Figure 10 on which we comment below.

Characteristic for the PEPA language is the choice to model parallel composition, or cooperation in the terminology of PEPA, scaled by the minimum of the so-called apparent rates. By doing so, PEPA’s strong equivalence becomes a congruence [Hillston(1996)]. Intuitively, the apparent rate $r_a(P)$ of an action $a$ for a process $P$ is the sum of the rates of all possible $a$-executions for $P$. The apparent rate $r_a(P)$ can easily be defined recursively on the structure of $P$ (see [Hillston(1996)] Definition 3.3.1) for details); accordingly, in the sequel we will refer to $r_a(P)$ as the ‘syntactic’ apparent rate. When considering the parallel composition $P \parallel Q$, with cooperation set $A$, an action $a$ occurring in $A$ has to be performed by both $P$ and $Q$. The rate of such an execution is governed by the slowest, on average, of the two processes in this respect (one cannot take the slowest process per sample, because such an operation cannot be expressed as an exponential distribution in general). Thus $r_a(P \parallel Q)$ for $a \in A$ is the minimum $\min\{r_a(P), r_a(Q)\}$. Now, if $P$ schedules an execution of $a$ with rate $r_1$ and $Q$ schedules a transition of $a$ with rate $r_2$, in the minimal apparent rate scheme the combined execution yields the action $a$ with rate $r_1 \cdot r_2 \cdot \text{arf}(P, Q)$. Here, the ‘syntactic’ scaling factor $\text{arf}(P, Q)$, the apparent rate factor, is defined by

$$\text{arf}(P, Q) = \frac{\min\{r_a(P), r_a(Q)\}}{r_a(P) \cdot r_a(Q)}$$

assuming $r_a(P), r_a(Q) > 0$, otherwise $\text{arf}(P, Q) = 0$. Organising the product $r_1 \cdot r_2 \cdot \text{arf}(P, Q)$ differently as $r_1 / r_a(P) \cdot r_2 / r_a(Q) \cdot \min\{r_a(P), r_a(Q)\}$ we see that for $P \parallel Q$ the minimum of the apparent rates $\min\{r_a(P), r_a(Q)\}$ is adjusted by the relative probabilities $r_1 / r_a(P)$ and $r_2 / r_a(Q)$ for executing $a$ by $P$ and $Q$, respectively.

The FuTS semantics of PEPA has been proposed originally in [De Nicola et al.(2011)]. The definition of the transition relation is recalled in Figure 10. The set of labels involved is $\Delta_\mathcal{A}$ defined by $\Delta_\mathcal{A} = \{ \delta_a | a \in \mathcal{A} \}$. In the context of FuTS semantics considered in this paper, we conventionally use the special symbol $\delta$ for denoting a random delay with a negative exponential distribution.
symbol $\delta_a$ denotes the duration of the execution of the action $a$ (assuming such a duration be random, exponentially distributed). The underlying semiring for the simple FuTS for PEPA is the semiring $\mathbb{R}_{\geq 0}$ of non-negative reals.

**Definition 8** The FuTS $S_{\text{pepa}} = (\mathcal{P}_{\text{pepa}}, \rightarrow_{\text{pepa}})$ over $\Delta_{\mathcal{A}}$ and $\mathbb{R}_{\geq 0}$ has its transition relation given by the rules of Figure 10.

We discuss the rules of Figure 10. The FuTS semantics provides $\text{nil} \rightarrow_{\text{pepa}} [\text{nil}]_{\mathbb{R}_{\geq 0}}$, for every action $a$, with $[\text{nil}]_{\mathbb{R}_{\geq 0}}$ the 0-function of $\mathbb{R}_{\geq 0}$. Therefore we have $\theta_{\text{pepa}}(\text{nil})(\delta_a)(P') = 0$ for every $a \in \mathcal{A}$ and $P' \in \mathcal{P}_{\text{pepa}}$, or, in standard terminology, nil has no transition. For the rated action prefix $(a, \lambda)$ we distinguish two cases: (i) execution of the prefix in rule (RAPF1); (ii) no execution of the prefix in rule (RAPF2). In the case of rule (RAPF1) the label $\delta_a$ signifies that the transition involves the execution of the action $a$. The continuation $[P \mapsto \lambda]$ is the function that assigns the rate $\lambda$ to the process $P$. All other processes are assigned 0, i.e. the zero-element of the semiring $\mathbb{R}_{\geq 0}$. In the second case, rule (RAPF2), for labels $\delta_b$ with $b \neq a$, we do have a state-to-function transition, but it is a degenerate one. The two rules for the prefix, in particular having the ‘null-continuation’ rule (RAPF2), support the unified treatment of the choice operator in rule (CHO) and the parallel operator in rules (PAR1) and (PAR2). The treatment of constants is as usual.

Note the semantic sum of functions $\mathcal{P} + \mathcal{D}$ replacing the syntactic sum in $P + Q$ and the semantic product $\mathcal{P} \bowtie \mathcal{D}$ used in rules (PAR1) and (PAR2) and, we recall, is defined as follows, for all $R \in \mathcal{P}_{\text{pepa}}$:

$$(\mathcal{P} \bowtie \mathcal{D})(R) = \begin{cases} \mathcal{P}(R_1) \cdot \mathcal{D}(R_2) & \text{if } R = R_1 \bowtie R_2 \text{ for some } R_1, R_2 \in \mathcal{P}_{\text{pepa}} \\ 0 & \text{otherwise} \end{cases}$$

Note that the syntactic construct $\bowtie$ of PEPA is trivially injective. Regarding the parallel operator $\bowtie$, with respect to some cooperation set $A \subseteq \mathcal{A}$ there are again two rules. Now the distinction is between interleaving and synchronisation. In the case of a label $\delta_a$ involving an action $a$ not in the subset $A$, either the $P$-operand or the $Q$-operand of $P \bowtie Q$ makes progress. For example, the effect of the pattern $\mathcal{P} \bowtie D_Q$ is that the value $\mathcal{P}(P') \cdot 1$ is assigned to a process $P' \bowtie Q$, the value $\mathcal{P}(P') \cdot 0 = 0$ to a process $P' \bowtie Q'$ for all $Q' \neq Q$, and the value 0 for a process not of the form $P' \bowtie Q'$. Here, as in all other rules, the right-hand sides of the transitions only involve functions in $\mathcal{F}S(\mathcal{P}_{\text{pepa}}, \mathbb{R}_{\geq 0})$ and operators on them.

For the synchronization case of the parallel construct, assuming $P \nrightarrow_{\text{pepa}} \mathcal{P}$ and $Q \nrightarrow_{\text{pepa}} \mathcal{D}$, the ‘semantic’ scaling factor $\text{arf}(\mathcal{P}, \mathcal{D})$ is applied to $\mathcal{P} \bowtie \mathcal{D}$. This scaling factor defined for functions in $\mathcal{F}S(\mathcal{P}_{\text{pepa}}, \mathbb{R}_{\geq 0})$, is, very much similar to its ‘syntactic’ counterpart, given by $\text{arf}(\mathcal{P}, \mathcal{D}) = \frac{\min \{ \mathcal{P} \bowtie \mathcal{D}, \mathcal{D} \bowtie \mathcal{P} \}}{\mathcal{P} \bowtie \mathcal{D}}$ provided $\mathcal{P}, \mathcal{D} > 0$, and $\text{arf}(\mathcal{P}, \mathcal{D}) = 0$ otherwise. For a process $R = R_1 \bowtie R_2$ we obtain the value $\text{arf}(\mathcal{P}, \mathcal{D}) \cdot (\mathcal{P} \bowtie \mathcal{D})(R_1 \bowtie R_2) = \text{arf}(\mathcal{P}, \mathcal{D}) \cdot \mathcal{P}(R_1) \cdot \mathcal{D}(R_2)$.

The following lemma establishes the relationship between the ‘syntactic’ and ‘semantic’ apparent rate factors defined on processes and on continuation functions, respectively.

**Lemma 12** Let $P \in \mathcal{P}_{\text{pepa}}$ and $a \in \mathcal{A}$. Suppose $P \nrightarrow_{\text{pepa}} \mathcal{P}$. Then $r_a(P) = \mathcal{P}$.

The proof of the lemma is straightforward (relying on the obvious definition of $r_a(P)$, omitted above, which can be found in [Hillston (1996)]). It is also easy to prove, by guarded induction, that the FuTS $S_{\text{pepa}}$ given by Definition 8 is total and deterministic. So, it is justified to write $S_{\text{pepa}} = (\mathcal{P}_{\text{pepa}}, \theta_{\text{pepa}})$. We use $\simeq_{\text{pepa}}$ to denote the bisimilarity induced by $S_{\text{pepa}}$. 

ASCENS
\[ (\text{RAPF}) \quad (a, \lambda).P \xrightarrow{a,\lambda} P \]
\[ (\text{CHO1}) \quad P \xrightarrow{a,\lambda} P + Q \xrightarrow{\pi} P' \]
\[ (\text{CHO2}) \quad Q \xrightarrow{\pi} P' \]
\[ (\text{PAR1a}) \quad P \xrightarrow{a,\lambda} P' \quad a \notin A \]
\[ (\text{PAR1b}) \quad Q \xrightarrow{a,\lambda} Q' \quad a \notin A \]
\[ (\text{PAR2}) \quad P \xrightarrow{a,\lambda} P' \quad Q \xrightarrow{a,\lambda} Q' \quad a \in A \quad \lambda = \text{arf}(P, Q) \cdot \lambda_1 \cdot \lambda_2 \]

Figure 11: Standard semantics for PEPA.

**Lemma 13** The FuTS \( S_{\text{pepa}} \) is total and deterministic. \( \Box \)

**Example** To illustrate the ease to deal with multiplicities in the FuTS semantics, consider the PEPA processes \( P_1 = (a, \lambda).P \) and \( P_2 = (a, \lambda).P + (a, \lambda).P \) for some \( P \in \mathcal{P}_{\text{pepa}} \). We have that \( P_1 \xrightarrow{\delta_a} \lambda \) by rule (RAPF1), but \( P_2 \xrightarrow{\delta_a} \lambda \) by rule (RAPF1) and rule (CHO). The latter makes us to compute \( [ P \xrightarrow{\lambda} ] + [ P \xrightarrow{\lambda} ] \), which equals \( [ P \xrightarrow{\lambda} ] \). Thus, in particular we have \( P_1 \nsubseteq S_{\text{pepa}} P_2 \). Intuitively it is clear that, in general we cannot have \( P + P \sim P \) for any reasonable quantitative process equivalence ~ in the Markovian setting. Having twice as many \( a \)-labelled transitions, the average number for \((a, \lambda).P + (a, \lambda).P \) of executing the action \( a \) per time unit is double the average of executing \( a \) for \((a, \lambda).P \).

The standard operational semantics of PEPA \cite{Hillston1996, Hillston2005} is given in Figure 11. The transition relation \( \xrightarrow{\text{pepa}} \subseteq \mathcal{P}_{\text{pepa}} \times (\mathcal{A} \times \mathbb{R}_{\geq 0}) \times \mathcal{P}_{\text{pepa}} \) is the least relation satisfying the rules. For a proper treatment of the rates, the transition relation is considered as a multi-transition system, where also the number of possible derivations of a transition \( P \xrightarrow{a,\lambda} P' \) matters. We stress that such bookkeeping is not needed in the FuTS-approach at all. In rule (PAR2) we use the ‘syntactic’ apparent rate factor for PEPA processes.

The so-called conditional transition rate \( q[P, C, a] \) of a PEPA-process for a subset of processes \( C \subseteq \mathcal{P}_{\text{pepa}} \) and \( a \in \mathcal{A} \) is given by (see, e.g. \cite{Hillston1996, Hillston2005}):

\[ q[P, C, a] = \sum_{Q \in C} \sum_{\lambda} \{ P \xrightarrow{a,\lambda} Q \}. \]

Here, \( \{ P \xrightarrow{a,\lambda} Q \} \) is the multiset of transitions \( P \xrightarrow{a,\lambda} Q \) and \( \{ \lambda \} \) \( P \xrightarrow{a,\lambda} Q \) is the multiset of all \( \lambda \)'s involved.

The multiplicity of \( P \xrightarrow{a,\lambda} Q \) is to be interpreted as the number of different ways the transition can be derived using the rules of Figure 11. We are now ready to define PEPA’s notion of strong equivalence \cite{Hillston1996, Hillston2005}.

**Definition 9** An equivalence relation \( R \subseteq \mathcal{P}_{\text{pepa}} \times \mathcal{P}_{\text{pepa}} \) is called a strong equivalence if

\[ q[P_1, [Q]_R, a] = q[P_2, [Q]_R, a] \]

for all \( P_1, P_2 \in \mathcal{P}_{\text{pepa}} \) such that \( R(P_1, P_2) \), all \( Q \in \mathcal{P}_{\text{pepa}} \) and all \( a \in \mathcal{A} \). Two processes \( P_1, P_2 \in \mathcal{P}_{\text{pepa}} \) are strongly equivalent if \( R(P_1, P_2) \) for a strong equivalence \( R \), notation \( P_1 \sim_{\text{pepa}} P_2 \).

The next lemma couples, for a PEPA-process \( P \), an action \( a \) and a function \( \mathcal{P} \in \mathcal{TS}(\mathcal{P}_{\text{pepa}}, \mathbb{R}_{\geq 0}) \), the evaluation \( \mathcal{P}(P) \) with respect to the FuTS-semantics to the cumulative rate for \( P \) of reaching \( P' \) by a transition involving the label \( a \) in the standard operational semantics.

---

\(^1\)In \cite{Hillston1996} strong equivalence is denoted by \( \simeq \); in this paper, we use \( \sim_{\text{pepa}} \), instead, for notational uniformity.
Lemma 14 Let \( P \in \mathcal{P}_{\text{pepa}} \) and \( a \in \mathcal{A} \). Suppose \( P \xrightarrow{\delta_a} \mathcal{P} \). The following holds: \( \mathcal{P}(P') = \sum \parallel \lambda \mid P \xrightarrow{a \lambda} P' \parallel \) for all \( P' \in \mathcal{P}_{\text{pepa}} \).

Proof Guarded induction on \( P \). We only treat the cases for the parallel composition. Note, the operation \( \triangleright_{\text{pepa}} : \mathcal{P}_{\text{pepa}} \times \mathcal{P}_{\text{pepa}} \rightarrow \mathcal{P}_{\text{pepa}} \) with \( \triangleright_{\text{pepa}}(P_1, P_2) = P_1 \triangleright_{\text{pepa}} P_2 \) is injective. Recall, for \( \mathcal{P}_1, \mathcal{P}_2 \in \mathcal{FS}(\mathcal{P}_{\text{pepa}}, \mathbb{R}_{\geq 0}) \), we have \( \mathcal{P}_1 \triangleright_{\text{pepa}} \mathcal{P}_2(P_1 \triangleright_{\text{pepa}} P_2) = \mathcal{P}_1(P_1) \cdot \mathcal{P}_2(P_2) \).

Suppose \( a \notin \mathcal{A} \). Assume \( P_1 \xrightarrow{\delta_a} \mathcal{P}_1, P_2 \xrightarrow{\delta_a} \mathcal{P}_2, P_1 \triangleright_{\text{pepa}} P_2 \xrightarrow{\delta_a} \mathcal{P} \). We distinguish three cases. Case (I), \( P' = P'_1 \triangleright_{\text{pepa}} P'_2, P'_1 \neq P_1 \). Then we have

\[
\sum \parallel \lambda \mid P_1 \triangleright_{\text{pepa}} P'_2 \parallel \xrightarrow{a \lambda} P' \\
= \sum \parallel \lambda \mid P'_1 \parallel \xrightarrow{a \lambda} P'_2 \parallel \hspace{1cm} \text{by rule (PAR1a)}
\]

\[
= \mathcal{P}_1(P'_1) \cdot \mathbf{D}_{P'_2}(P_2)
\]

\[
= (\mathcal{P}_1 \triangleright_{\text{pepa}} \mathcal{P}_2)(P_1 \triangleright_{\text{pepa}} P_2)
\]

\[
= \mathcal{P}(P')
\]

Case (II), \( P' = P_1 \triangleright_{\text{pepa}} P'_2, P'_2 \neq P_2 \); similar.

Case (III), \( P' = P_1 \triangleright_{\text{pepa}} P'_2 \). Then we have:

\[
\sum \parallel \lambda \mid P_1 \triangleright_{\text{pepa}} P'_2 \parallel \xrightarrow{a \lambda} P' \\
= (\sum \parallel \lambda \mid P_1 \parallel \xrightarrow{a \lambda} P_2 \parallel) + \sum \parallel \lambda \mid P_2 \parallel \xrightarrow{a \lambda} P'_2 \parallel \hspace{1cm} \text{by rules (PAR1a) and (PAR1b)}
\]

\[
= \mathcal{P}_1(P_1) + \mathcal{P}_2(P_2)
\]

\[
= (\mathcal{P}_1 \triangleright_{\text{pepa}} \mathcal{P}_2)(P_1 \triangleright_{\text{pepa}} P_2)
\]

\[
= \mathcal{P}(P')
\]

Suppose \( a \in \mathcal{A} \). Assume \( P_1 \xrightarrow{\delta_a} \mathcal{P}_1, P_2 \xrightarrow{\delta_a} \mathcal{P}_2, P_1 \triangleright_{\text{pepa}} P_2 \xrightarrow{\delta_a} \mathcal{P} \). Without loss of generality, \( P' = P'_1 \triangleright_{\text{pepa}} P'_2 \) for suitable \( P'_1, P'_2 \in \mathcal{P}_{\text{pepa}} \).

\[
\sum \parallel \lambda \mid P_1 \triangleright_{\text{pepa}} P'_2 \parallel \xrightarrow{a \lambda} P' \\
= \sum \parallel \text{arf}(P_1, P_2) \cdot \lambda_1 \cdot \lambda_2 \mid P_1 \xrightarrow{a \lambda_1} P'_1, P_2 \xrightarrow{a \lambda_2} P'_2 \parallel \hspace{1cm} \text{by rule (PAR2)}
\]

\[
= \text{arf}(P_1, P_2) \cdot (\sum \parallel \lambda_1 \mid P_1 \xrightarrow{a \lambda_1} P'_1 \parallel) \cdot (\sum \parallel \lambda_2 \mid P_2 \xrightarrow{a \lambda_2} P'_2 \parallel) \hspace{1cm} \text{by distributivity}
\]

\[
= \text{arf}(P_1, P_2) \cdot \mathcal{P}_1(P'_1) \cdot \mathcal{P}_2(P'_2)
\]

\[
= \text{arf}(\mathcal{P}_1 \cdot \mathcal{P}_2) \cdot \mathcal{P}_1(P'_1) \cdot \mathcal{P}_2(P'_2)
\]

\[
= \mathcal{P}(P')
\]

The other cases are simpler and omitted here.

With the lemma in place we can prove the following correspondence result for \( S_{\text{pepa}} \)-bisimilarity and strong equivalence as given by Definition.
**Theorem 15** For any two PEPA-processes \( P_1, P_2 \in \mathcal{P}_{\text{pepa}} \) the following holds: \( P_1 \sim_{\text{pepa}} P_2 \) iff \( P_1 \sim_{\text{FuTS}} P_2 \).

**Proof** Let \( R \) be an equivalence relation on \( \mathcal{P}_{\text{pepa}} \). Choose \( P, Q \in \mathcal{P}_{\text{pepa}} \) and \( a \in \mathcal{A} \). Suppose \( P \overset{\delta_a}{\rightarrow}_{\text{pepa}} P \). Thus \( \theta_{\text{pepa}}(P)(\delta_a) = \mathcal{P} \). We have

\[
q[P, [Q]_R, a] = \sum Q' \in [Q]_R \sum \lambda \mid P \overset{a,\lambda}{\rightarrow}_{\text{pepa}} Q' \quad \text{by definition } q[P, [Q]_R, a]
\]

\[
= \sum Q' \in [Q]_R \mathcal{P}(Q') \quad \text{by Lemma 14}
\]

\[
= \sum Q' \in [Q]_R \theta_{\text{pepa}}(P)(a)(Q') \quad \text{by definition } \theta_{\text{pepa}}
\]

Therefore, for PEPA-processes \( P_1 \) and \( P_2 \) it holds that \( q[P_1, [Q]_R, a] = q[P_2, [Q]_R, a] \) for all \( Q \in \mathcal{P}_{\text{pepa}} \), \( a \in \mathcal{A} \) iff \( \sum Q' \in [Q]_R \theta_{\text{pepa}}(P_1)(a)(Q') = \sum Q' \in [Q]_R \theta_{\text{pepa}}(P_2)(a)(Q') \) for all \( Q \in \mathcal{P}_{\text{pepa}} \), \( a \in \mathcal{A} \). Thus, the equivalence relation \( R \) is a strong equivalence iff \( R \) is an \( S_{\text{pepa}} \)-bisimulation, from which the theorem follows.

In view of our general correspondence result Theorem [7] the above theorem shows that PEPA’s strong equivalence \( \sim_{\text{pepa}} \) is a behavioural equivalence, viz. the behavioural equivalence on the FuTS \( S_{\text{pepa}} \), when seen as a \( V_{\mathcal{A}, X}^{\mathcal{R}_{>0}} \)-coalgebra, which, in turn, coincides with the associated coalgebraic bisimilarity.

In other words, letting \( V_{\text{pepa}} \) abbreviate \( V_{\mathcal{A}, X}^{\mathcal{R}_{>0}} \) the following equalities hold:

\[
\sim_{\text{pepa}} = \approx_{S_{\text{pepa}}} = \approx_{V_{\text{pepa}}} = \approx_{S_{\text{FuTS}}}
\]

Thus, PEPA’s standard, FuTS, behavioural and coalgebraic semantics coincide.

## 7 FuTS Semantics of IML

In this section we provide a FuTS semantics for a relevant part of IML, the language of Interactive Markov Chains [Hermanns(2002)]. IMCs for short. IMCs are automata that combine two types of transitions: interactive transitions that involve the execution of actions and Markovian transitions that represent the progress of time governed by exponential distributions. As a consequence, IMCs embody both non-deterministic and stochastic behaviour. System analysis using IMCs proves to be a powerful approach because of the orthogonality of qualitative and quantitative dynamics, their logical underpinning and tool support, cf. [Bohnenkamp et al.(2006)] [Hermanns & Katoen(2010)] and [Bozga et al.(2012)]. A number of behavioural equivalences, both strong and weak, are available for IMCs [Eisenbraut et al.(2010a)]. In our treatment here, discussing a subset we call IMLs, we do not deal with internal \( \tau \)-steps and focus on strong bisimilarity. The FuTS semantics we consider in the sequel has been proposed in [De Nicola et al.(2011)].

**Definition 10** The set \( \mathcal{P}_{\text{iml}} \) of IML processes is given by the grammar below:

\[
P ::= \text{nil} \mid a.P \mid \lambda.P \mid P + P \mid \parallel A \mid P \mid X
\]

where \( a \) ranges over the set of actions \( \mathcal{A} \), \( \lambda \) over \( \mathbb{R}_{>0} \), \( A \) over the set of finite subsets of \( \mathcal{A} \) and \( X \) over the set of constants \( X \).

We assume the same notation and guardedness requirements for constant definition and usage as in Section [5] for PEPA.

In line with the discussion above, in IML there are separate prefix constructions for actions \( a.P \) and for time-delays \( \lambda.P \). No restriction is imposed on the alternative and parallel composition of processes. For example, we have the process \( a.\lambda.nil + \mu.b.nil \) in IML. It should be noted that for IMCs actions are considered to take no time.
Lemma 16 The formal semantics of TR 09: Coalgebraic Bisimulation of FuTSs (V1) January 16, 2013

The formal semantics of FuTS is total and deterministic. Below we use $S_{\text{iml}} = (\mathcal{P}_{\text{iml}}, \rightarrow_1, \rightarrow_2)$ over the label sets $\mathcal{A}$ and $\Delta = \{\delta\}$ and the semirings $\mathbb{B}$ and $\mathbb{R}_{\geq 0}$ with transition relations $\rightarrow_1 \subseteq \mathcal{P}_{\text{iml}} \times \mathcal{A} \times \mathcal{FS}(\mathcal{P}_{\text{iml}}, \mathbb{B})$ and $\rightarrow_2 \subseteq \mathcal{P}_{\text{iml}} \times \Delta \times \mathcal{FS}(\mathcal{P}_{\text{iml}}, \mathbb{R}_{\geq 0})$ defined as the least relations satisfying the rules of Figure 12.

To accommodate for action-based and delay-related transitions, the FuTS $S_{\text{iml}}$ is non-simple, having the two state-to-function relations $\rightarrow_1$ and $\rightarrow_2$. Actions $a \in \mathcal{A}$ decorate $\rightarrow_1$, the special symbol $\delta$ decorates $\rightarrow_2$. Note rule (APF3) and rule (RPF1) involve the null-functions of $\mathbb{R}_{\geq 0}$ and of $\mathbb{B}$, respectively, to express that a process $a.P$ does not trigger a delay and a process $\lambda.P$ does not execute an action. For the parallel construct $\parallel$, interleaving applies both for non-synchronised actions $a \notin A$ as well as for delays (but not mixed). Therefore, rule (PAR1) pertains to both $\rightarrow_1$ and $\rightarrow_2$, with $\alpha$ ranging over $\mathcal{A} \cup \Delta$. The same holds for non-deterministic choice, rule (CHO), and constants, rule (CON). Finally, IML does not provide synchronization of delays in the parallel construct. Rule (PAR2) only concerns the transition relation $\rightarrow_1$. In rule (PAR1), for clarity, we decorated the characteristic functions, writing $D_i$, for $i = 1, 2$, for $D_i = [P \mapsto \text{true}]$ in $\mathcal{FS}(\mathcal{P}_{\text{iml}}, \mathbb{B})$ and $D_i = [P \mapsto 1]$ in $\mathcal{FS}(\mathcal{P}_{\text{iml}}, \mathbb{R}_{\geq 0})$. We recall that for all $R \in \mathcal{P}_{\text{iml}}$:

$$(\mathcal{P} \parallel A \mathcal{D})(R) = \begin{cases} \mathcal{P}(R_1) \cdot \mathcal{D}(R_2), & \text{if } R = R_1 \parallel A R_2 \text{ for some } R_1, R_2 \in \mathcal{P}_{\text{iml}} \\ 0, & \text{otherwise} \end{cases}$$

where $\cdot$ is the product in $\mathbb{R}_{\geq 0}$ whenever $\mathcal{P}, \mathcal{D} \in \mathcal{FS}(\mathcal{P}_{\text{iml}}, \mathbb{R}_{\geq 0})$ and is the logical conjunction $\land$ for $\mathcal{P}, \mathcal{D} \in \mathcal{FS}(\mathcal{P}_{\text{iml}}, \mathbb{B})$.

Example Assume $X := a.\lambda.b.X$ and $Y := a.\mu.b.Y$. Put $A = \{a, b\}$. Then we have

$X \parallel A Y \rightarrow_1 [a.\lambda.b.X \parallel A \mu.b.Y \mapsto \text{true}] \quad \lambda.b.X \parallel A b.Y \rightarrow_2 [b.X \parallel A b.Y \mapsto \lambda]$\n
$\quad b.X \parallel A b.Y \rightarrow_1 [X \parallel A Y \mapsto \text{true}] \quad b.X \parallel A \mu.b.Y \rightarrow_2 [b.X \parallel A b.Y \mapsto \mu]$\n
It is not difficult to verify that $S_{\text{iml}}$ is a total and deterministic FuTS. Below we use $S_{\text{iml}} = (\mathcal{P}_{\text{iml}}, \theta_1, \theta_2)$ and write $\approx_{S_{\text{iml}}}$ for the associated bisimilarity.

Lemma 16 The FuTS $S_{\text{iml}}$ is total and deterministic.

The standard SOS semantics of IML [Hermanns(2002)] is given in Figure 13 involving the transition relations

$\rightarrow \subseteq \mathcal{P}_{\text{iml}} \times \mathcal{A} \times \mathcal{P}_{\text{iml}} \quad \text{and} \quad \rightarrow \subseteq \mathcal{P}_{\text{iml}} \times \mathbb{R}_{\geq 0} \times \mathcal{P}_{\text{iml}}$
Below we will use the functions $T$ and $R$ based on $\to$ and $\xrightarrow{}$, cf. \cite{HermannsKatoen2010}. We have $T: \mathcal{P}_{\text{iml}} \times \mathcal{A} \times 2^\mathcal{P}_{\text{iml}} \to \mathbb{B}$ given by $T(P, a, C) = \text{true}$ if the set $\{ P' \in C \mid P \xrightarrow{a} P' \}$ is non-empty, for all $P \in \mathcal{P}_{\text{iml}}, a \in \mathcal{A}$ and any subset $C \subseteq \mathcal{P}_{\text{iml}}$. For $R: \mathcal{P}_{\text{iml}} \times \mathcal{P}_{\text{iml}} \to \mathbb{R}_{\geq 0}$ we put $R(P, P') = \sum_{\lambda} \| P \xrightarrow{\lambda} P' \|$, for $P \in \mathcal{P}_{\text{iml}}, C \subseteq \mathcal{P}_{\text{iml}}$. For IML we have the following notion of strong bisimulation \cite{Hermanns2002, HermannsKatoen2010} that we will compare with the notion of bisimulation associated with the FuTS $S_{\text{iml}}$.

**Definition 12** An equivalence relation $R \subseteq \mathcal{P}_{\text{iml}} \times \mathcal{P}_{\text{iml}}$ is called a strong bisimulation for IML if, for all $P_1, P_2 \in \mathcal{P}_{\text{iml}}$ such that $R(P_1, P_2)$, it holds that

- for all $a \in \mathcal{A}$ and $Q \in \mathcal{P}_{\text{iml}}$: $T(P_1, a, [Q]_R) \iff T(P_2, a, [Q]_R)$

- for all $Q \in \mathcal{P}_{\text{iml}}$: $R(P_1, [Q]_R) = R(P_2, [Q]_R)$.

Two processes $P_1, P_2 \in \mathcal{P}_{\text{iml}}$ are called strongly bisimilar if $R(P_1, P_2)$ for a strong bisimulation $R$ for IML, notation $P_1 \approx_{\text{iml}} P_2$.

To establish the correspondence of FuTS bisimilarity $\approx_{\text{iml}}$ for $S_{\text{iml}}$ of Definition 11 and strong bisimilarity $\approx_{\text{iml}}$ for IML, we need to connect the state-to-function relation $\rightarrow_1$ and the transition relation $\rightarrow$ as well as the state-to-function relation $\rightarrow_2$ and the transition relation $\rightarrow$.

**Lemma 17** (a) Let $P \in \mathcal{P}_{\text{iml}}$ and $a \in \mathcal{A}$. If $P \xrightarrow{a}_1 \mathcal{P}$ then $P \xrightarrow{a} P' \iff \mathcal{P}(P') = \text{true}$.  

(b) Let $P \in \mathcal{P}_{\text{iml}}$. If $P \xrightarrow{\delta}_2 \mathcal{P}$ then $\sum_{\lambda} \| P \xrightarrow{\lambda} P' \| = \mathcal{P}(P')$.  

**Proof** (a) Guarded induction. Let $a \in \mathcal{A}$. We treat the typical cases $\lambda, P$ and $P_1 \parallel_A P_2$ for $a \notin A$.

Case $\lambda, P$. Suppose $\lambda, P \xrightarrow{a}_1 \mathcal{P}$. Then we have $\mathcal{P} = [\| \mathbb{B}]$. Both $\lambda, P \xrightarrow{a} P'$ for no $P' \in \mathcal{P}_{\text{iml}}$, as no transition is provided in $\rightarrow$, and $\mathcal{P}(P') = \text{false}$ by definition of $[\| \mathbb{B}]$, for all $P' \in \mathcal{P}_{\text{iml}}$.

Case $P_1 \parallel_A P_2$, $a \notin A$. Suppose $P_1 \xrightarrow{a}_1 \mathcal{P}_1$, $P_2 \xrightarrow{a}_1 \mathcal{P}_2$ and $P_1 \parallel_A P_2 \xrightarrow{a}_1 \mathcal{P}$. Then it holds that $\mathcal{P} = (\mathcal{P}_1 \parallel_A \mathcal{D}_{P_1}) + (\mathcal{D}_{P_1} \parallel_A \mathcal{P}_2)$. Recall, for $Q \in \mathcal{P}_{\text{iml}}$ and $\mathcal{D}_{Q} \in \mathcal{T}(\mathcal{P}_{\text{iml}}, \mathbb{B})$, $\mathcal{D}_{Q}(Q') = \text{true}$ if
\( Q' = Q \), for \( Q' \in \mathcal{P}_{iml} \). We have

\[
P_1 \parallel A P_2 \xrightarrow{a} P' \\
\leftrightarrow (P_1 \xrightarrow{a} P'_1 \lor P' = P'_1 \parallel A P_2) \lor (P_2 \xrightarrow{a} P'_2 \lor P' = P_1 \parallel A P'_2)
\]

by analysis of \( \xrightarrow{\cdot} \)

\[
\leftrightarrow (\mathcal{P}_1(P'_1) = \text{true} \lor P' = P'_1 \parallel A P_2) \lor (\mathcal{P}_2(P'_2) = \text{true} \lor P' = P_1 \parallel A P'_2)
\]

by the induction hypothesis

\[
\leftrightarrow (\mathcal{P}_1(P'_1) \cdot D_{P_2}(P_2) = \text{true} \lor P' = P'_1 \parallel A P_2) \lor (D_{P_1}(P_1) \cdot \mathcal{P}_2(P'_2) = \text{true} \lor P' = P_1 \parallel A P'_2)
\]

by definition of \( D_{P_1} \) and \( D_{P_2} \)

\[
\leftrightarrow ((\mathcal{P}_1 \parallel A D_{P_2})(P'_1 \parallel A P_2) = \text{true} \lor (D_{P_1} \parallel A \mathcal{P}_2)(P'_2) = \text{true})
\]

by definition of \( \parallel A \)

\[
\leftrightarrow (\mathcal{P}_1 \parallel A D_{P_2})(P') = \text{true} \lor (D_{P_1} \parallel A \mathcal{P}_2)(P') = \text{true}
\]

by definition of \( + \) on \( \mathcal{FS}(\mathcal{P}_{iml}, \mathbb{B}) \)

\[
\leftrightarrow \mathcal{P}(P') = \text{true}
\]

The other cases are standard or similar and easier.

(b) Guarded induction. We treat the cases for \( \mu. P \) and \( P_1 \parallel A P_2 \). Case \( \mu. P \). Assume \( P \xrightarrow{\delta} P' \). Suppose \( \mu P' \). Then it holds that \( P \) admits a single \( \xrightarrow{\cdot} \)-transition, viz. \( P \xrightarrow{\mu} P' \). Thus we have

\[
\sum \parallel A P \xrightarrow{\cdot} P' = \mu = \{ P' \in P, P' = \mu \parallel P' \} = \mathcal{P}(P')
\]

Suppose \( \mu P' \) for some \( P' \neq P \). Then we have

\[
\sum \parallel A P \xrightarrow{\cdot} P' = 0 = \{ P'' \in P, P'' = \mu \parallel P'' \} = \mathcal{P}(P')
\]

Case \( P_1 \parallel A P_2 \). Assume \( P_1 \xrightarrow{\delta} P_1', P_2 \xrightarrow{\delta} P_2 \) and \( P_1 \parallel A P_2 \xrightarrow{\delta} P_2 \). It holds that \( \mathcal{P} = (\mathcal{P}_1 \parallel A D_{P_2}) + (D_{P_1} \parallel A \mathcal{P}_2) \). We calculate

\[
\sum \parallel A P_1 \parallel A P_2 \xrightarrow{\cdot} P' = \sum \parallel A P_1 \xrightarrow{\cdot} P'_1, P' = P'_1 \parallel A P_2 + \sum \parallel A P_2 \xrightarrow{\cdot} P'_2, P' = P_1 \parallel A P'_2
\]

by analysis of \( \xrightarrow{\cdot} \)

\[
= (\text{if } P' = P'_1 \parallel A P_2 \text{ then } \sum \parallel A P_1 \xrightarrow{\cdot} P'_1 \text{ else 0 end}) +
\]

\[
(\text{if } P' = P_1 \parallel A P'_2 \text{ then } \sum \parallel A P_2 \xrightarrow{\cdot} P'_2 \text{ else 0 end})
\]

by induction hypothesis for \( P_1 \) and \( P_2 \)

\[
= (\mathcal{P}_1 \parallel A D_{P_2})(P') + (D_{P_1} \parallel A \mathcal{P}_2)(P')
\]

by definition of \( \parallel A \), \( D_{P_1}, D_{P_2} \) and \( + \) on \( \mathcal{FS}(\mathcal{P}_{iml}, \mathbb{B}) \)

\[
= \mathcal{P}(P')
\]

The remaining cases are left to the reader.

We are now in a position to relate \( \text{FuTS} \) bisimilarity and standard strong bisimilarity for \( \text{IML} \).

**Theorem 18** For any two processes \( P_1, P_2 \in \mathcal{P}_{iml} \) it holds that \( P_1 \simeq_{\text{iml}} P_2 \) iff \( P_1 \sim_{\text{iml}} P_2 \).
\textbf{Proof} Let $R$ be an equivalence relation on $\mathcal{P}_{\text{iml}}$. Pick $P \in \mathcal{P}_{\text{iml}}$, $a \in \mathcal{A}$ and choose any $Q \in \mathcal{P}_{\text{iml}}$. Suppose $P \xrightarrow{\alpha} \mathcal{P}$. Then we have

\[
\begin{align*}
\mathbf{T}(P, a, [Q]_R) &\iff \exists Q' \in [Q]_R : P \xrightarrow{\alpha} Q' \quad \text{by definition of } \mathbf{T} \\
&\iff \exists Q' \in [Q]_R : \mathcal{P}(Q') = \text{true} \quad \text{by Lemma}\ [7] \\
&\iff \sum Q \in [Q]_R \theta_1(P)(a)(Q) = \text{true} \quad \text{by definition of } \theta_1
\end{align*}
\]

Note, summation in $\mathbb{B}$ is disjunction. Likewise, on the quantitative side, we have

\[
\begin{align*}
\mathbf{R}(P, [Q]_R) &= \sum Q \in [Q]_R \sum \lambda : P \xrightarrow{\lambda} Q' \quad \text{by definition of } \mathbf{R} \\
&= \sum Q \in [Q]_R \mathcal{P}(Q') \quad \text{by Lemma}\ [7] \\
&= \sum Q \in [Q]_R \theta_2(P)(\delta)(Q) \quad \text{by definition of } \theta_2
\end{align*}
\]

Combining the equations, we conclude that a strong bisimulation for IML is also an $S_{\text{iml}}$-bisimulation for the $\text{FuTS} S_{\text{iml}}$, and vice versa. From this the theorem follows.

Again, as a corollary of the theorem above, we have for IML that its notion of strong bisimilarity $P_1 \sim_{\text{iml}} P_2$ is coalgebraically underpinned, as it coincides, calling to Theorem [7] once more, with behavioural equivalence on the $\text{FuTS} S_{\text{iml}}$, when seen as a $\mathcal{V}_{(\mathbb{B}, \times)}^{(\mathbb{A}, \delta)}$-coalgebra, which, in turn, coincides with the associated coalgebraic bisimilarity. In other words, letting $\mathcal{V}_{\text{iml}}$ abbreviate $\mathcal{V}_{(\mathbb{B}, \times)}^{(\mathbb{A}, \delta)}$, the following equalities hold:

\[
\sim_{\text{iml}} = \equiv_{S_{\text{iml}}} = \equiv_{\mathcal{V}_{\text{iml}}} = \sim_{\mathcal{V}_{\text{iml}}}
\]

Thus, IML’s standard, $\text{FuTS}$, behavioural and coalgebraic semantics coincide.

\section{Discussion}

Above we have focused on the treatment with $\text{FuTS}$s of action prefix in the setting of an elementary process language, stochastic prefix in the setting of PEPA and their mixing in the setting of IML and studied the positioning of the associated notion of process strong bisimulation equivalences. The semirings involved are the booleans $\mathbb{B}$ and non-negative reals $\mathbb{R}_{\geq 0}$. The orthogonality of $\text{FuTS}$s allows for superposition of various state-to-function transition relations, allowing e.g. to mingle with discrete deterministic time, as we will sketch below. Also, when considering repeated application of functors $\mathcal{F}S(\cdot, \mathcal{R})$ more complex state-to-function transition system can be defined, for example to deal with so-called Markov automata [Eisenbraut et al.(2010a)] [Eisenbraut et al.(2010b)].

With the help of the semiring of non-negative integers $\mathbb{N}$, a $\text{FuTS}$-style semantics can be given to discrete deterministic time processes where time elapses by the ticking of the clock. For example, [Aldini et al.(2010)] discusses a small process language, called TPC, involving the prefix construct $(n).P$, with $n \in \mathbb{N}$, $n > 0$, expressing that the process $P$ is to be executed after $n$ time steps. A $\text{FuTS}$ for TPC can be of type $(\mathcal{P}_{\text{TPC}}, \xrightarrow{1})$ over the label sets $\mathcal{A}$ and $\{\sqrt{\cdot}\}$—where $\mathcal{A}$ is a set of actions as before and the special symbol $\sqrt{\cdot}$ denotes a fixed discrete deterministic delay representing progress in time—and the semirings $\mathbb{B}$ and $\mathbb{N}$. We have $\xrightarrow{1} \subseteq \mathcal{P}_{\text{TPC}} \times \mathcal{A} \times \mathcal{F}S(\mathcal{P}_{\text{TPC}}, \mathbb{B})$ and $\xrightarrow{2} \subseteq \mathcal{P}_{\text{TPC}} \times \{\sqrt{\cdot}\} \times \mathcal{F}S(\mathcal{P}_{\text{TPC}}, \mathbb{N})$. The relevant rules involving the timed prefix construct are

\[
\begin{align*}
\text{(TPF1)} &\quad \frac{a \in \mathcal{A}}{(n).P \xrightarrow{\alpha} [(n)P]} \\
\text{(TPF2)} &\quad \frac{P \xrightarrow{\sqrt{\alpha}} \mathcal{P}}{(n).P \xrightarrow{\sqrt{n}} n; [P \xrightarrow{n}] + (n + \mathcal{P})}
\end{align*}
\]

The first timed prefix rule (TPF1) expresses that a timed prefix cannot perform an action (immediately). The second time prefix rule (TPF2) combines a possible evolution over time of the process $P$
into its continuation $\mathcal{P}$ with the elapse of the prefix. Note, the continuation in the conclusion of rule (TPF2) is a sum of three parts, viz. $[n; P]$, $[P \mapsto n]$, $(n + \mathcal{P})$. The mappings $[n; P]$ and $(n + \mathcal{P})$ are given by

$$
[n; P](Q) = \begin{cases} 
  m & \text{if } 0 < m < n \text{ and } Q = (n-m).P \\
  0 & \text{otherwise}
\end{cases}
$$

$$(n + \mathcal{P})(Q) = \begin{cases} 
  n + \mathcal{P}(Q) & \text{if } \mathcal{P}(Q) > 0 \\
  0 & \text{otherwise}
\end{cases}$$

Time progress taking fewer steps than $n$ is covered by the continuation $[n; P]$. For $m$ strictly between 0 and $n$, after $m$ time steps there remains $(n-m).P$ to be executed. After exactly $n$ time steps, $P$ is to be executed. After more than $n$ time steps, say $n + m$ time steps, process $Q$ is to be executed if $\mathcal{P}(Q) = m$, for $m > 0$.

The rules for the choice and parallel construct of TPC make use of corresponding operations on $\mathcal{FS}(\mathcal{TPC}, \mathbb{N})$ such that so-called time-determinism and time-continuity principles are respected. Again a total and deterministic FuTS is obtained this way. Also, along the lines of the correspondence proofs for PEPA and IML, it can be shown that the notion of discrete-time bisimulation of [Aldini et al. (2010)] Baeten & Middelburg (2002) and the notion of bisimulation for the FuTS sketched above as well as the associated notion of behavioural equivalence coincide. Thus, also deterministic time can be handled with FuTSs. Note, as the semiring $\mathbb{N}$ does not possess multiplicative inverses, we cannot appeal to Theorem [F] to connect to coalgebraic equivalence in this case.

Markov automata, as proposed in [Eisentraut et al. (2010a)] Eisentraut et al. (2010b), combine non-deterministic and probabilistic behaviour with stochastic time. The combination of non-deterministic and probabilistic behaviour provided by Markov automata can be easily achieved, at the linguistic level by means of a combination of a standard choice operator, $\cdot$, with the following probabilistic extension of action prefix: $a.[p_1 :: P_1 \sqcap \ldots \sqcap p_h :: P_h]$ with $a \in \mathcal{A}$, the set of actions, and $h > 0$, $p_1, \ldots, p_h \in (0, 1]$ such that $p_1 + \cdots + p_h \leq 1$. The syntactic construct $[p_1 :: P_1 \sqcap \ldots \sqcap p_h :: P_h]$ denotes the sub-distribution $\mathcal{D}(p_1::P_1\sqcap\ldots\sqcap p_h::P_h)$ over processes defined by

$$
\mathcal{D}(p_1::P_1\sqcap\ldots\sqcap p_h::P_h) = \sum_{i=1}^{h} [P_i \mapsto p_i]
$$

The intuitive meaning is then obvious: process $a.[p_1 :: P_1 \sqcap \ldots \sqcap p_h :: P_h]$ performs action $a$ and then behaves as process $P$ with probability $\mathcal{D}(p_1::P_1\sqcap\ldots\sqcap p_h::P_h)(P)$.

The amalgamation of action prefix and probabilistic choice leads to a nesting of the functors involved. Now, transitions labelled by actions go from processes to sets of discrete sub-distributions. In fact, following the FuTS approach, one encounters transitions of the form $\mapsto_1 \subseteq \mathcal{P}_{\mathcal{MA}} \times \mathcal{A} \times \mathcal{FS}(\mathcal{SDistr}(\mathcal{P}_{\mathcal{MA}}), \mathbb{B})$ to deal with the non-deterministic/probabilistic aspect of the language as well as transitions of the form $\mapsto_2 \subseteq \mathcal{P}_{\mathcal{MA}} \times \Delta \times \mathcal{FS}(\mathcal{P}_{\mathcal{MA}}, \mathbb{R}_{\geq 0})$ to handle stochastic delays, as we have already seen in the previous section. Here, $\mathcal{SDistr}(\cdot)$ is the functor associating finite sub-distributions to a set, dealing with functions similar to the FuTS functors. More concretely, for the combined action and probabilistic choice prefix, we have the rules

$$(\text{APF1}) \quad a.[p_1 :: P_1 \sqcap \ldots \sqcap p_h :: P_h] \mapsto_1 [\mathcal{D}(p_1::P_1\sqcap\ldots\sqcap p_h::P_h)] \mapsto_{\text{true}}$$

$$(\text{APF2}) \quad \frac{b \neq a}{a.[p_1 :: P_1 \sqcap \ldots \sqcap p_h :: P_h] \mapsto_1 [\mathbb{B}] \mapsto_{b}}$$

$$(\text{APF3}) \quad a.[p_1 :: P_1 \sqcap \ldots \sqcap p_h :: P_h] \mapsto_2 [\mathbb{R}_{\geq 0}]$$

We expect that all coalgebraic reasoning will hold true for such nesting of functors. In particular, we claim for a generalised notion of (deterministic and total) FuTS, viz. coalgebras $(X, \theta)$ of a functor $(\mathcal{F}_1 \circ \cdots \circ \mathcal{F}_n)^L$ or a product of such functors, where $\mathcal{F}_i = \mathcal{FS}(\cdot, \mathcal{R}_i)$ for some semiring $\mathcal{R}_i$, $i = 1 \ldots n$,
that the associated notion of \(\text{FuTS}\) bisimilarity coincides with behavioural equivalence, and, for semirings having multiplicative inverses and meeting the zero-sum property (as discussed in Section 4), with coalgebraic bisimilarity as well.

9 Concluding remarks

Total and deterministic state-to-function labeled transition systems, \(\text{FuTS}\), are a convenient instrument to express the operational semantics of both qualitative and quantitative process languages. In this paper we have discussed the notion of bisimilarity that arises from a \(\text{FuTS}\), possibly involving multiple transition relations, from a coalgebraic perspective. For \(\text{FuTS}\) models of two process languages based on prominent stochastic process algebras we related the induced notion of bisimulation to the standard equivalences, thus providing these equivalence with a coalgebraic underpinning. The main technical contribution of our paper is a correspondence result, Theorem 7, that relates bisimilarity of a \(\text{FuTS} \ S\) to behavioural equivalence of the functor associated with \(S\).

It is noted in [Bonchi et al.(2011)], in the context of weighted automata, that in general the type of functors \(\mathcal{F}S(\cdot, \mathcal{R})\) may not preserve weak pullbacks and, therefore, the notions of coalgebraic bisimilarity and of behavioural equivalence may not coincide. A counter example is provided, cf. [Bonchi et al.(2011), Section 2.2]. Essential for the construction of the counter-example, in their setting, is the fact that the sum of non-zero weights may add to weight 0. The same phenomenon prevents a general proof, along the lines of [de Vink & Rutten(1999)], for coalgebraic bisimilarity and \(\text{FuTS}\) bisimilarity to coincide. In the construction of a mediating morphism, going from \(\text{FuTS}\) bisimulation to coalgebraic bisimulation a denominator may be zero, hence a division undefined, in case the sum over an equivalence class cancels out. In the concrete case for [Klin & Sassone(2008)], although no detailed proof is provided there, this will not happen with \(\mathbb{R}_{\geq 0}\) as underlying semiring. Here we propose to consider semirings which admit a (right) multiplicative inverse for non-zero elements, and satisfy the so-called zero-sum property, stating that for a sum \(x = x_1 + \cdots + x_n\) it holds that \(x = 0\) iff \(x_i = 0\) for all \(i = 1, \ldots, n\). We have shown, Theorem 9, that, when the semirings involved enjoy these properties, weak pullbacks are preserved by the associated functor. Therefore, coalgebraic bisimilarity and behavioural equivalence are the same. As a consequence, under conditions which are met by the \(\text{SPC}\)'s proposed in the literature, we have that \(\text{FuTS}\)-bisimilarity, behavioural equivalence and coalgebraic bisimilarity coincide.

For two prototypical stochastic process languages based on \(\text{PEPA}\) and on \(\text{IMC}\) we have shown that the notion of strong equivalence and strong bisimilarity associated with these calculi, coincides with the notion of bisimilarity of the corresponding \(\text{FuTS}\). Using these \(\text{FuTS}\)s as a stepping stone, the correspondence result bridges between the concrete notion of bisimulation for \(\text{PEPA}\) and \(\text{IML}\), and the coalgebraic notions of behavioural equivalence and coalgebraic bisimilarity. Hence, from this perspective, the concrete notions are seen as the natural strong equivalence to consider. Obviously, classical strong bisimilarity [Milner(1980) Park(1981)] and bisimilarity for \(\text{FuTS}\) over \(\mathbb{B}\) coincide. Also, strong bisimulation of [Hillston(1996)] involving, apart from the usual transfer conditions, the comparison of state information, viz. the apparent rates, can be treated with \(\text{FuTS}\). Again the two notions of equivalence coincide. Finally, we gave an account how languages based on discrete deterministic time as well as those where stochastic time is integrated with discrete probability and with non-determinism can be easily treated in the \(\text{FuTS}\) framework. Future research needs to reveal under what algebraic conditions of the semirings, or similar structures, or the coalgebraic conditions on the format of the functors involved standard bisimulation, \(\text{FuTS}\)-bisimulation, coalgebraic bisimulation and behavioural equivalence will amount to similar identifications also for the above mentioned mod-
els. In particular, the study of nested functors (i.e. compositions of functors) seems to be promising.

Acknowledgments The authors are grateful to Rocco De Nicola, Fabio Gadducci, Daniel Gebler, Michele Loreti and Jan Rutten for fruitful discussions on the subject and useful suggestions. DL and MM acknowledge support by EU Project n. 257414 Autonomic Service-Components Ensembles (ASCENS). This research has been conducted while EV was spending a sabbatical leave at the CNR/ISTI. EV gratefully acknowledges the hospitality and support during his stay in Pisa.

References


