

Probabilistic I/O Automata: Theories of Two Equivalences*

Eugene W. Stark¹, Rance Cleaveland², Scott A. Smolka^{1**}

¹ Department of Computer Science
State University of New York
at Stony Brook
Stony Brook, NY 11794 USA

² Department of Computer Science
University of Maryland
College Park, Maryland 20742 USA

Abstract. Working in the context of a process-algebraic language for Probabilistic I/O Automata (PIOA), we study the notion of *PIOA behavior equivalence* by obtaining a complete axiomatization of its equational theory and comparing the results with a complete axiomatization of a more standard equivalence, *weighted bisimulation*. The axiomatization of behavior equivalence is achieved by adding to the language an operator for forming *convex combinations* of terms.

Keywords: stochastic process algebras; process equivalences; continuous-time Markov chains; equational theories; complete axiomatizations

1 Introduction

In previous work [SCS03], we presented a process-algebraic language, motivated by the *probabilistic I/O automaton* model, that provides a compositional formalism for defining continuous-time Markov chains (CTMCs). The constructs in our language are similar to those in other “Markovian process algebra” languages that have been studied by a number of other researchers (see [HH02] for a survey), especially EMPA [BDG98]. In our language, we classify transitions as either *output* (“active”) transitions or *input* (“passive”) transitions. Output transitions, which can occur spontaneously, have associated positive *rates*. Rates are dimensional quantities with units of 1/time, which are regarded as the parameters of exponential probability distributions. When multiple output transitions are available for a process, the choice between them is made probabilistically by a “race policy” semantics: an exponentially distributed random future time is chosen for the occurrence of each transition (using the associated rate as the

* This research was supported in part by the National Science Foundation under Grant CCR-9988155 and the Army Research Office under Grants DAAD190110003 and DAAD190110019. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, the Army Research Office, or other sponsors.

** Authors’ E-mail addresses: {stark,sas}@cs.sunysb.edu, rance@cs.umd.edu

parameter of the distribution) and the transition for which the earliest time is chosen “wins the race” and becomes the next transition to occur. Input transitions, which can occur for a process only in conjunction with a similarly labeled output transition performed by its environment, have associated positive *weights*, which are dimensionless. When multiple input transitions labeled by the same action are available, the choice between them is made probabilistically on the basis of their proportionate weights.

In this paper, we consider a fragment of our language having the following syntax, where Act is a set of *actions*, variables a, b, c, \dots are used to range over Act , and variables t, u, v, \dots are used to range over process terms:

$$\text{nil}_I \mid \langle a?_w \rangle t \mid \langle b!_r \rangle t \mid t + t' \mid t \parallel_{O'} t'$$

The informal meaning of the constructs is as follows:

- nil_I denotes a process that passively accepts an input from the set $I \subseteq Act$, assigning such an input a weight of 1, and then continues to behave as nil_I .
- $\langle a?_w \rangle t$ denotes an *input-prefixed* process that can accept an input $a \in Act$ with positive *weight* w and then become the process t .
- $\langle b!_r \rangle t$ denotes an *output-prefixed* process that can spontaneously perform output action $b \in Act$ with positive *rate* r and then become the process denoted by t .
- $t + t'$ denotes a *choice* between alternatives offered by t and t' .
- $t \parallel_{O'} t'$ denotes the *parallel composition* of t and t' . Here O and O' are disjoint sets of output actions *controlled by* t and t' , respectively.

Synchronization of actions, which occurs between the components of a parallel composition, is restricted to the input/input and input/output cases only. Output/output synchronization is not permitted. This seems to be the simplest version of action synchronization that has an intuitively meaningful stochastic interpretation. As our goal is to understand the relationship between input and output in the simplest possible setting, we do not complicate the language with immediate actions, priorities, or other extraneous constructs.

The standard notions of process equivalence in the context of stochastic process algebra are based on variants of *probabilistic bisimulation* [LS92], which is closely related to the concept of *lumpability* in the theory of Markov chains. A typical example of such an equivalence is *Markovian bisimulation* [Hil96], in which terms regarded as equivalent are required to have the same aggregate transition rate, and which is such that equivalent terms have identical total transition probabilities to each equivalence class of terms for each action. In this paper we use *weighted bisimulation*, which uses the same fundamental idea and covers the cases of weight-labeled and rate-labeled transitions.

Behavior equivalence is an alternative to weighted bisimulation equivalence that we have studied in earlier papers. This equivalence is strictly coarser than weighted bisimulation equivalence, but still substitutive with respect to the process algebraic operations listed above. The original motivation of behavior equivalence was as a testing equivalence, and in this context a full-abstractness result

was established in [WSS97]. The original definitions were reformulated in subsequent papers as our understanding of behavior equivalence improved. In [Sta03] we were able to compare weighted bisimulation equivalence and behavior equivalence by viewing them both as certain “invariant” equivalences on formal *linear combinations* of process terms, rather than as equivalences on individual terms. We showed, roughly: (1) that weighted bisimulation equivalence can be characterized as the largest invariant equivalence on combinations of terms that is in a sense generated by equations between individual terms, (2) that behavior equivalence can be characterized as the largest invariant equivalence on combinations of terms that in a sense separates terms having distinct aggregate rates, and (3) that behavior equivalence is strictly coarser than weighted bisimulation equivalence, even when restricted to individual terms.

For example, the following intuitively reasonable equation between terms in our language holds for behavior equivalence but not for weighted bisimulation equivalence:

$$\langle b!_r \rangle (\langle c!_{\pi s} \rangle t + \langle d!_{(1-\pi)s} \rangle u) = \langle b!_{\pi r} \rangle \langle c!_s \rangle t + \langle b!_{(1-\pi)r} \rangle \langle d!_s \rangle u$$

where π can be any value in the interval $(0, 1)$. Intuitively, both sides above can perform the output b with the same aggregate rate r . After doing so, the term on the left-hand side evolves to the derivative term $\langle c!_{\pi s} \rangle t + \langle d!_{(1-\pi)s} \rangle u$, which can do output c with rate πs and output d with rate $(1 - \pi)s$, for an aggregate rate of s . In contrast, there is no individual term that expresses the derivative of the right-hand side after output b has been performed. The best we can do is to think of this derivative as a *probability distribution* that assigns probability π to term $\langle c!_s \rangle t$ and probability $1 - \pi$ to term $\langle d!_s \rangle t$. Intuitively, there is no observable difference between such a probability distribution and the individual term $\langle c!_{\pi s} \rangle t + \langle d!_{(1-\pi)s} \rangle u$, which explains why the original equation is a reasonable one to expect.

Consideration of the preceding example suggests that an axiomatization of behavior equivalence might be achieved if we augment the language with an explicit notation for expressing convex combinations of terms; for example: $\langle c!_s \rangle t \pi \oplus_{1-\pi} \langle d!_s \rangle u$. We would then be able to express the equivalence between the derivatives of the left and right-hand sides of the equation above as follows:

$$\langle c!_{\pi s} \rangle t + \langle d!_{(1-\pi)s} \rangle u = \langle c!_s \rangle t \pi \oplus_{1-\pi} \langle d!_s \rangle u.$$

In fact, an axiomatization of behavior equivalence can be achieved in this way and the details are the subject of the present paper. A key point, which took us a long time to discover, is that we cannot permit the the formation of combinations $t \pi \oplus_{1-\pi} u$ for arbitrary terms t and u . Rather, we must require as a condition of well-formedness that terms t and u have an *identical aggregate rate*, which then becomes the aggregate rate of the combined term. Failing to impose this requirement results in the possibility of having “terms” that do not have unique aggregate rates, which produces seemingly insurmountable complications in the semantics and axiomatization. Another detail that required some care to work

out properly concerns keeping track of the “types” of terms, by which we mean the sets of input and output actions in which a term is required to participate.

As a result of our investigation, we have further clarified our understanding of behavior equivalence and its relationship to weighted bisimulation equivalence. Perhaps the simplest way to summarize what we have learned is to compare the normal form used in the proofs of completeness for the axiomatization of weighted bisimulation equivalence with that used in the proof for behavior equivalence. Employing \sum -notation in a standard way and (for the moment) ignoring special cases that arise with empty summations, the following is a generic normal form for a term with respect to weighted bisimulation equivalence:

$$\sum_{i=1}^m \langle a_i ?_{w_i} \rangle t_i + \sum_{j=1}^n \langle b_j !_{r_j} \rangle t_j$$

In the above, the t_i and t_j are recursively required to be normal forms. Moreover, it is required that for no distinct i and i' do we have both $a_i = a_{i'}$ and t_i equivalent to $t_{i'}$ and for no distinct j and j' do we have both $b_j = b_{j'}$ and t_j equivalent to $t_{j'}$. Thus, w_i is the aggregate weight of a_i -transitions to the equivalence class of t_i , and r_j is the aggregate rate of b_j -transitions to the equivalence class of t_j .

In contrast, a generic normal form for a term with respect to behavior equivalence is the following:

$$\sum_{a \in I} \sum_{s \in R_a} \langle a ?_{w_{a,s}} \rangle t_{a,s} + \sum_{b \in O} \sum_{s \in R_b} \langle b !_{r_{b,s}} \rangle t_{b,s},$$

where each set R_a and R_b is nonempty and each term $t_{a,s}$ and $t_{b,s}$ is required recursively to be a normal form with aggregate rate s . The main point here is that, once input a has been chosen, there is a unique derivative term $t_{a,s}$ for each aggregate rate in the set R_a , and once output b has been chosen, there is a unique derivative term $t_{b,s}$ for each aggregate rate in the set R_b . Terms $t_{a,s}$ and $t_{a,s'}$ cannot be equivalent for distinct values of s because they have distinct aggregate rates. Similar considerations hold for $t_{b,s}$ and $t_{b,s'}$. A normal form for behavior equivalence is thus also a normal form for weighted bisimulation equivalence, but not conversely. So, the essential difference between weighted bisimulation equivalence and behavior equivalence is that the former will in general draw distinctions between terms based on the existence of multiple derivatives having the same aggregate rate, whereas the latter will not.

Note that, although the operator $\pi \oplus_{1-\pi}$ does not appear in the normal form for behavior equivalence, achieving a reduction to normal form will in general require passing through terms in which explicit use is made of this operator.

The remainder of the paper is organized as follows: In Section 2, we summarize the basic definitions pertaining to our process-algebraic language and its semantics. In Section 3, we define the notion of weighted bisimulation equivalence for our language and present a sound and complete set of axioms for this equivalence. In Section 4, we define the notion of behavior equivalence, extend the language with the convex combination operator $\pi \oplus_{1-\pi}$ discussed above,

$$\begin{array}{c}
\text{nil}_I : I/I \Rightarrow \emptyset \\
\\
\frac{t : J/J \Rightarrow O \quad a \in J}{\langle a?_w \rangle t : \{a\}/J \Rightarrow O} \quad \frac{t : J/J \Rightarrow O \quad b \notin J}{\langle b!_r \rangle t : \emptyset/J \Rightarrow O \cup \{b\}} \\
\\
\frac{t : I_t/J \Rightarrow O_t \quad u : I_u/J \Rightarrow O_u}{t + u : I_t \cup I_u/J \Rightarrow O_t \cup O_u} \\
\\
\frac{t : I_t/I_t \Rightarrow O_t \quad u : I_u/I_u \Rightarrow O_u \quad I = (I_t \cup I_u) \setminus (O_t \cup O_u)}{t \text{ }_{O_t} \parallel_{O_u} u : I/I \Rightarrow O_t \cup O_u} \\
\\
\frac{t : I/J \Rightarrow O \quad O \subseteq O' \quad O' \cap J = \emptyset}{t : I/J \Rightarrow O'}
\end{array}$$

Fig. 1. Type-Inference Rules

present a sound and complete set of axioms for behavior equivalence in the extended language, and sketch the main ideas of the completeness proof. Although we include the parallel composition construct in the language defined in section 2, the results of Sections 3 and 4 concern only the \parallel -free fragment. We hope to extend our results to include parallel composition in a future paper.

Though we give here complete definitions for all important notions, space limitations force us to omit almost all proofs from this paper. The interested reader can find the omitted proofs in the full version [SCS06] available online.

2 Basic Definitions

2.1 Types

As detailed in our previous paper [SCS03], our PIOA language is equipped with a set of rules for inferring *typing judgements* of the form $t : I/J \Rightarrow O$, where I , J , and O are sets of actions. We write $\vdash t : \phi$ to assert that a typing judgement $t : \phi$ is inferable. A term t is *well-typed* if $\vdash t : \phi$ for some ϕ . Let $\text{Proc}(I/J \Rightarrow O)$ denote the set of all terms t such that $\vdash t : I/J \Rightarrow O$.

Intuitively, a typing judgement $t : I/J \Rightarrow O$ asserts that I is a set of actions for which input transitions are guaranteed to be enabled at the first step of t , that J is a set of actions for which input transitions are guaranteed to be enabled at all steps of t after the first, and O is a set of actions that includes at least all the outputs that may be produced by t (but which may be larger). The primary purpose of the typing system is to identify those terms that are *input-enabled*, in order to rule out the formation of parallel compositions involving non-input-enabled terms. Non-input-enabled terms are required in the language to permit the building up of sets of alternatives using $+$. The reason why only input-enabled terms are permitted in parallel compositions is that we do not wish to allow stochastically unclear situations in which one component in a system is attempting to perform an output with a definite rate, but is inhibited from doing so by another component that will not accept that action as an input.

Figure 1 presents the type-inference rules applicable to the language fragment we consider here. We have included an additional “weakening” rule (the last rule), which was not present in our previous paper. The purpose of the weakening rule is to ensure that if $\vdash t : I/J \Rightarrow O$ then also $\vdash t : I/J \Rightarrow O'$ for all $O' \supseteq O$ such that $O' \cap J = \emptyset$; this is a useful property that did not hold of the typing system in our previous paper.

Proposition 1. *If $\vdash t : I/J \Rightarrow O$ for some I, J , and O , then*

1. $I \subseteq J$ and $J \cap O = \emptyset$.
2. *There exists \hat{O} such that $\vdash t : I/J \Rightarrow \hat{O}$, and such that whenever $\vdash t : I'/J' \Rightarrow O'$ then $I' = I, J' = J$, and $O' \supseteq \hat{O}$.*

A technical issue with our PIOA language is that of “native” versus “non-native” actions. If $t \in \text{Proc}(I/J \Rightarrow O)$, then actions $e \in J \cup O$ are called *native* to t and actions outside this set are called *non-native*. Intuitively, native actions are those in which t must participate and non-native actions are those that t ignores. This distinction is important because if t has no transition for a particular output action in which it must participate, then that action is inhibited from occurring, whereas if t ignores an action then it may occur freely. Note that whether an action is considered native or non-native depends on our having fixed a particular type $I/J \Rightarrow O$ inferable for t . All such types have the same input sets I and J , but the output sets O may differ. Thus, in the sequel it will be necessary for us to parameterize certain notions by the particular output set on which they depend.

2.2 Transition Semantics

In our previous paper, we gave structural operational semantics rules that defined the transitions that could be taken by terms in our language. Though our present purposes do not require a full presentation of the transition semantics given in our previous paper, we do need a notation for the aggregate *weight* or *rate* $\Delta_e^O(t, v)$ of e -labeled transitions from t to v .

Suppose $t \in \text{Proc}(I/J \Rightarrow O)$. Define $\Delta_e^O(t, v)$ as follows: If $e \notin J \cup O$ (non-native case), then $\Delta_e^O(t, v) = 1$ if $v = t$, and $\Delta_e^O(t, v) = 0$ otherwise. If $e \in J \cup O$ (native case), then

1. $\Delta_e^O(\text{nil}_I, v) = \begin{cases} 1, & \text{if } e \in I \text{ and } v = \text{nil}_I. \\ 0, & \text{otherwise.} \end{cases}$
2. $\Delta_e^O(\langle a?_w \rangle t, v) = \begin{cases} w, & \text{if } e = a \text{ and } v = t \\ 0, & \text{otherwise.} \end{cases}$
3. $\Delta_e^O(\langle b!_r \rangle t, v) = \begin{cases} r, & \text{if } e = b \text{ and } v = t \\ 0, & \text{otherwise.} \end{cases}$
4. $\Delta_e^O(t + u, v) = \Delta_e^O(t, v) + \Delta_e^O(u, v)$.

$$5. \Delta_e^O(t \text{ }_{O_i} \parallel_{O_u} u, v) = \begin{cases} \Delta_e^{O \setminus O_u}(t, t') \cdot \Delta_e^{O \setminus O_i}(u, u'), & \text{if } v = t' \text{ }_{O_i} \parallel_{O_u} u' \\ 0, & \text{otherwise.} \end{cases}$$

In the sequel, if \mathcal{C} is a set of terms, then $\Delta_e^O(t, \mathcal{C})$ will be an abbreviation for the sum $\sum_{v \in \mathcal{C}} \Delta_e^O(t, v)$, which is always finite.

It is important for us that inferable types are preserved under transitions. Formally, we have the following result, which was stated in our previous paper and remains true in the presence of the weakening rule.

Proposition 2. *Suppose $t \in \text{Proc}(I/J \Rightarrow O)$. If $\Delta_e^O(t, u) \neq 0$, then $u \in \text{Proc}(J/J \Rightarrow O)$. In particular, $\text{Proc}(J/J \Rightarrow O)$ is closed under transitions and terms in $\text{Proc}(I/J \Rightarrow O)$ reach $\text{Proc}(J/J \Rightarrow O)$ after one transition.*

A term $t \in \text{Proc}(J/J \Rightarrow O)$ is called *input-stochastic* if for all $e \in J$ we have $\sum_{v \in \text{Proc}(J/J \Rightarrow O)} \Delta_e^O(t, v) = 1$.

3 Weighted Bisimulation

In this paper, we discuss weighted bisimulation equivalence primarily for the purposes of comparison with behavior equivalence. Modulo minor differences in the formal setup, the properties of this equivalence are standard, and have been established before by other authors (*e.g.* [HR94]). Consequently, in this section we simply give the basic definitions and state the results briefly without proof.

A *weighted bisimulation* on $\text{Proc}(J/J \Rightarrow O)$ is an equivalence relation R on $\text{Proc}(J/J \Rightarrow O)$ such that the following condition is satisfied:

- Whenever $t R t'$ then for all actions e and all equivalence classes \mathcal{C} of R we have $\Delta_e^O(t, \mathcal{C}) = \Delta_e^O(t', \mathcal{C})$.

Terms t and t' in $\text{Proc}(I/J \Rightarrow O)$ are defined to be *weighted bisimulation equivalent* if there exists a weighted bisimulation relation R on $\text{Proc}(J/J \Rightarrow O)$ such that for all actions e and all equivalence classes \mathcal{C} of R we have $\Delta_e^O(t, \mathcal{C}) = \Delta_e^O(t', \mathcal{C})$. In this case we write $t \underset{O}{\sim} t'$ (there is no need to mention the input sets I and J which are uniquely determined by t and t').

Without giving a formal statement (it is similar to the one given later for behavior equivalence), we comment at this point that a substitutivity result can be established for weighted bisimulation equivalence, which makes an equational axiomatization feasible.

3.1 Axioms

Axioms for weighted bisimulation equivalence are shown in Table 1. In axiom (nil-fold), we have used the summation notation $\sum_{a \in I} \langle a?_1 \rangle \text{nil}_I$ in an obvious way. The soundness of axioms (choice-comm) and (choice-assoc) will permit us in the sequel to manipulate this summation notation in the conventional fashion without further comment.

Lemma 1. *The axioms shown in Table 1 are sound for weighted bisimulation equivalence.*

$t + \text{nil}_\emptyset = t$	(choice-unit)
$t + u = u + t$	(choice-comm)
$(t + u) + v = t + (u + v)$	(choice-assoc)
$\langle a?_p \rangle t + \langle a?_q \rangle t = \langle a?_{p+q} \rangle t$	(input-choice)
$\langle b!_r \rangle t + \langle b!_s \rangle t = \langle b!_{r+s} \rangle t$	(output-choice)
If $I \neq \emptyset$, then $\sum_{a \in I} \langle a?_1 \rangle \text{nil}_I = \text{nil}_I$	(nil-fold)

Table 1. Axioms for Weighted Bisimulation Equivalence

3.2 Completeness

We say that two terms are *identical up to permutation of sums* if they can be proved equivalent to each other using only axioms (choice-comm) and (choice-assoc).

Let the notions *input normal form*, *output normal form*, and *normal form* be defined mutually recursively as follows:

- An *input normal form* is a well-typed term u that is either nil_I for some $I \neq \emptyset$, or else has the form $\sum_{i=1}^m \langle a_i?_{p_i} \rangle t_i$, where we require that:
 1. Each t_i is a normal form.
 2. For no distinct i, i' do we have $a_i = a_{i'}$ and t_i identical to $t_{i'}$ up to permutation of sums.
 3. u is not an instance (up to permutation of sums) of the left-hand side of axiom (nil-fold).
- An *output normal form* is a well-typed term v that is either nil_\emptyset or else has the form $\sum_{j=1}^n \langle b_j!_{r_j} \rangle t_j$, where we require that:
 1. Each t_j is a normal form.
 2. For no distinct j, j' do we have $b_j = b_{j'}$ and t_j identical to $t_{j'}$ up to permutation of sums.
An output normal form is called *nontrivial* if it is not nil_\emptyset .
- A *normal form* is either an input normal form, an output normal form, or a sum $u + v$, where u is an input normal form and v is a nontrivial output normal form.

Lemma 2. Any $\|\text{-free}$ term t in $\text{Proc}(I/J \Rightarrow O)$ can be proved equivalent to a normal form using the axioms in Table 1.

Lemma 3. If normal forms t and t' in $\text{Proc}(I/J \Rightarrow O)$ are weighted bisimulation equivalent, then they are identical up to permutation of sums.

Theorem 1. The axioms in Table 1 are sound and complete for weighted bisimulation equivalence of $\|\text{-free}$ terms.

4 Behavior Equivalence

4.1 Behavior Maps

We now consider the theory of behavior equivalence. To define behavior equivalence, we need some auxiliary concepts. First is the notion of the *aggregate rate* $\text{rt}(t)$ of a term $t \in \text{Proc}(I/J \Rightarrow O)$. This is defined by: $\text{rt}(t) = \sum_{e \in O} \sum_{t'} \Delta_e^O(t, t')$. The following can then be established by structural induction:

- If t has the form nil_I or $\langle a?_w \rangle u$, then $\text{rt}(t) = 0$.
- If t has the form $\langle b!_r \rangle u$, then $\text{rt}(t) = r$.
- If t has the form $u + v$, then $\text{rt}(t) = \text{rt}(u) + \text{rt}(v)$.
- If t has the form $u \mathbin{O_u} \parallel_{O_v} v$, where u and v are input-stochastic (cf. Section 2.2), then $\text{rt}(t) = \text{rt}(u) + \text{rt}(v)$.

Next, we define a *rated action* to be a pair $\langle e, r \rangle \in \text{Act} \times [0, \infty)$. A *rated trace* is a finite sequence of rated actions. We use ϵ to denote the empty rated trace.

An *observable* is a mapping from rated traces to real numbers. We use Obs to denote the set of all observables. The *derivative* of an observable Φ by a rated action $\langle e, r \rangle$ is the observable $\langle e, r \rangle^{-1}\Phi$ defined by

$$(\langle e, r \rangle^{-1}\Phi)(\alpha) = \Phi(\langle e, r \rangle \alpha)$$

for all all rated traces α .

To each term t in $\text{Proc}(I/J \Rightarrow O)$ we associate a *behavior map* $\mathcal{B}_t^O : \text{Obs} \rightarrow \text{Obs}$ defined by induction on the length of an observation as follows:

1. $\mathcal{B}_t^O[\Phi](\epsilon) = \Phi(\epsilon)$.
2. $\mathcal{B}_t^O[\Phi](\langle e, r \rangle \alpha) = \sum_{u \in \text{Proc}(J/J \Rightarrow O)} \Delta_e^O(t, u) \cdot \mathcal{B}_u^O[\langle e, r + \text{rt}(t) \rangle^{-1}\Phi](\alpha)$

Terms t and t' in $\text{Proc}(I/J \Rightarrow O)$ are defined to be *behavior equivalent*, and we write $t \equiv t'$, if $\mathcal{B}_t^O = \mathcal{B}_{t'}^O$.

The following result gives a syntax-directed characterization of \mathcal{B}_t^O that is useful in proofs. Case (1) is concerned with non-native actions.

Lemma 4. *For all terms $t \in \text{Proc}(I/J \Rightarrow O)$, all observables Φ , all rated traces α and rated actions $\langle e, r \rangle$:*

1. $\mathcal{B}_t^O[\Phi](\langle e, r \rangle \alpha) = \mathcal{B}_t^O[\langle e, r + \text{rt}(t) \rangle^{-1}\Phi](\alpha)$, if $e \notin J \cup O$.
2. $\mathcal{B}_{\text{nil}_J}^O[\Phi](\langle e, r \rangle \alpha) = \begin{cases} \mathcal{B}_{\text{nil}_J}^O[\langle e, r \rangle^{-1}\Phi](\alpha), & \text{if } e \in J, \\ 0, & \text{if } e \in O. \end{cases}$
3. $\mathcal{B}_{\langle a?_w \rangle t}^O[\Phi](\langle e, r \rangle \alpha) = \begin{cases} w \cdot \mathcal{B}_t^O[\langle a, r \rangle^{-1}\Phi](\alpha), & \text{if } e = a \\ 0, & \text{if } e \in (J \cup O) \setminus \{a\}. \end{cases}$

4. $\mathcal{B}_{\langle b, s \rangle t}^O[\Phi](\langle e, r \rangle \alpha) = \begin{cases} s \cdot \mathcal{B}_t^O[\langle b, r + s \rangle^{-1} \Phi](\alpha), & \text{if } e = b \\ 0, & \text{if } e \in (J \cup O) \setminus \{b\}. \end{cases}$
5. $\mathcal{B}_{t+u}^O[\Phi](\langle e, r \rangle \alpha) = \mathcal{B}_t^O[\Phi](\langle e, r + \text{rt}(u) \rangle \alpha) + \mathcal{B}_u^O[\Phi](\langle e, r + \text{rt}(t) \rangle \alpha).$
6. $\mathcal{B}_{t \circ_t \parallel_{O_u} u}^O = \mathcal{B}_t^{O \setminus O_u} \circ \mathcal{B}_u^{O \setminus O_t} = \mathcal{B}_u^{O \setminus O_t} \circ \mathcal{B}_t^{O \setminus O_u}$, assuming t and u are input-stochastic.

Input-stochasticity is required in (6) in order to ensure that $\text{rt}(t' \circ_t \parallel_{O_u} u') = \text{rt}(t') + \text{rt}(u')$ for all derivatives t' of t and u' of u . Without this assumption, compositionality fails. Further discussion of behavior maps and their properties can be found in our previous papers [WSS97, Sta03, SCS03].

4.2 Combinations

As indicated in the introduction, in order to axiomatize behavior equivalence, we extend our language by adding a construct for forming (convex) *combinations* of terms. Specifically, we add an additional binary operator $\pi \oplus_{1-\pi}$, where the parameter π is a real number in the open interval $(0, 1)$.

The following typing rule applies to this new operator:

$$\frac{t : I/J \Rightarrow O \quad u : I/J \Rightarrow O \quad \text{rt}(t) = \text{rt}(u)}{t \pi \oplus_{1-\pi} u : I/J \Rightarrow O}$$

This rule requires that, for $t \pi \oplus_{1-\pi} u$ to be well-typed, terms t and u must have the same aggregate rate as well as a common type. In this case we extend the notion of aggregate rate by defining $\text{rt}(t \pi \oplus_{1-\pi} u)$ to be the common value $\text{rt}(t) = \text{rt}(u)$.

We formally extend the transition semantics Δ_e^O given in Section 2.2 to encompass terms containing $t \pi \oplus_{1-\pi} u$ by adding to the defining clauses given there the additional clause:

$$\Delta_e^O(t \pi \oplus_{1-\pi} u, v) = \pi \cdot \Delta_e^O(t, v) + (1 - \pi) \cdot \Delta_e^O(u, v).$$

Note that in making the extension we are implicitly re-interpreting the original clauses from Section 2.2 by allowing for the possibility of terms containing $\pi \oplus_{1-\pi}$. For example, we now have

$$\Delta_a^O(\langle a?_w \rangle (t \pi \oplus_{1-\pi} u), t \pi \oplus_{1-\pi} u) = w.$$

We similarly re-interpret the definition of \mathcal{B}_t^O given earlier in this section to allow for the possibility of terms containing $\pi \oplus_{1-\pi}$. These particular definitions are intuitively motivated by our desire for $t \pi \oplus_{1-\pi} u$ to represent a probabilistic choice between (or superposition of) t and u ; as considered, for example, in [And99]. Formally, we obtain the following result:

Lemma 5. *For all terms t, u in $\text{Proc}(I/J \Rightarrow O)$ such that $\text{rt}(t) = \text{rt}(u)$, for all observables Φ and all rated traces α we have:*

$$\mathcal{B}_{t \pi \oplus_{1-\pi} u}^O[\Phi](\alpha) = \pi \cdot \mathcal{B}_t^O[\Phi](\alpha) + (1 - \pi) \cdot \mathcal{B}_u^O[\Phi](\alpha).$$

Lemma 6. *Behavior equivalence is substitutive for input prefixing, output prefixing, choice, combination, and also for parallel composition of input-stochastic terms. That is, each of the following assertions holds for terms t and t' in $\text{Proc}(I/J \Rightarrow O)$ whenever all the terms mentioned are well-typed and the equivalences make sense:*

1. If $t \stackrel{O}{\equiv} t'$ then $\langle a?_w \rangle t \stackrel{O}{\equiv} \langle a?_w \rangle t'$.
2. If $t \stackrel{O}{\equiv} t'$ then $\langle b!_r \rangle t \stackrel{O}{\equiv} \langle b!_r \rangle t'$.
3. If $t \stackrel{O}{\equiv} t'$ then $t + u \stackrel{O}{\equiv} t' + u$ and $u + t \stackrel{O}{\equiv} u + t'$.
4. If $t \stackrel{O}{\equiv} t'$ then $t \pi_{\oplus 1-\pi} u \stackrel{O}{\equiv} t' \pi_{\oplus 1-\pi} u$ and $u \pi_{\oplus 1-\pi} t \stackrel{O}{\equiv} u \pi_{\oplus 1-\pi} t'$.
5. If $t \stackrel{O}{\equiv} t'$ then $t \circ_{O_u} u \stackrel{O}{\equiv} t' \circ_{O_u} u$ and $u \circ_{O_u} t \stackrel{O}{\equiv} u \circ_{O_u} t'$,
assuming $t, t',$ and u are input-stochastic.

4.3 Axioms

Axioms for behavior equivalence are shown in Table 2. Note that an equation is only regarded an axiom if all the terms involved are well-formed and the same type can be inferred for the left and right-hand sides. Particular care must be taken when using axioms (input-distr) and (output-distr), to see that these equations are never applied in such a way as to create combinations whose operands have different rates.

In contrast to the axioms for weighted bisimulation equivalence, the axioms for behavior equivalence expose some distinction between input and output. For example, comparison of axiom (input-comb) and (output-comb) reveals that in (output-comb) the two output actions are permitted to be distinct. This is not permitted in (input-comb), because in that case the right-hand side would never be well-typed. Also, the axiom (input-extract) exhibits a special property of input that is not shared by output. The content of axiom (interchange) is that the two types of sums commute freely with each other, subject only to the conditions on rates imposed by well-typedness.

Note that Table 2 includes all the axioms for weighted bisimulation equivalence, except for the axioms (input-choice) and (output-choice). However, it is not difficult to show that these axioms are derivable, so that all equations provable for weighted bisimulation equivalence are also provable for behavior equivalence.

Lemma 7. *The axioms in Table 2 are sound for behavior equivalence.*

Proof. We prove the case of (input-extract) to give the basic flavor of working with behavior maps. We must show that

$$\mathcal{B}_{\langle a?_{\pi p} \rangle t+(u \pi_{\oplus 1-\pi} v)}^O[\Phi](\alpha) = \mathcal{B}_{\langle (a?_p) t+u \rangle \pi_{\oplus 1-\pi} v}^O[\Phi](\alpha)$$

for all observables Φ and rated traces α . We proceed by induction on the length of the rated trace α . The basis case $\alpha = \epsilon$ is immediate, because by definition of behavior maps:

$$\mathcal{B}_{\langle a?_{\pi p} \rangle t+(u \pi_{\oplus 1-\pi} v)}^O[\Phi](\epsilon) = \Phi(\epsilon) = \mathcal{B}_{\langle (a?_p) t+u \rangle \pi_{\oplus 1-\pi} v}^O[\Phi](\epsilon).$$

$$\begin{aligned}
t + \text{nil}_\emptyset &= t && \text{(choice-unit)} \\
t + u &= u + t && \text{(choice-comm)} \\
(t + u) + v &= t + (u + v) && \text{(choice-assoc)}
\end{aligned}$$

$$\sum_{a \in I} \langle a?_1 \rangle \text{nil}_I = \text{nil}_I \quad \text{(nil-fold)}$$

$$\langle a?_p \rangle t + \langle a?_q \rangle u = \langle a?_{p+q} \rangle t \frac{p}{p+q} \oplus \frac{q}{p+q} \langle a?_{p+q} \rangle u \quad \text{(input-comb)}$$

$$\langle b!_r \rangle t + \langle c!_s \rangle u = \langle b!_{r+s} \rangle t \frac{r}{r+s} \oplus \frac{s}{r+s} \langle c!_{r+s} \rangle u \quad \text{(output-comb)}$$

$$t = t \pi \oplus_{1-\pi} t \quad \text{(comb-idemp)}$$

$$t \pi \oplus_{1-\pi} u = u \ 1-\pi \oplus_{\pi} t \quad \text{(comb-comm)}$$

$$(t \pi \oplus_{1-\pi} u) \rho \oplus_{1-\rho} v = t \sigma \oplus_{1-\sigma} (u \ \tau \oplus_{1-\tau} v), \\
\text{whenever } \pi\rho = \sigma \text{ and } (1-\rho) = (1-\sigma)(1-\tau). \quad \text{(comb-assoc)}$$

$$\langle a?_p \rangle t \ \pi \oplus_{1-\pi} \langle a?_p \rangle u = \langle a?_p \rangle (t \ \pi \oplus_{1-\pi} u) \quad \text{(input-distr)}$$

$$\langle b!_r \rangle t \ \pi \oplus_{1-\pi} \langle b!_r \rangle u = \langle b!_r \rangle (t \ \pi \oplus_{1-\pi} u) \quad \text{(output-distr)}$$

$$(t \ \pi \oplus_{1-\pi} w) + (u \ \pi \oplus_{1-\pi} v) = (t + u) \ \pi \oplus_{1-\pi} (w + v) \quad \text{(interchange)}$$

$$\langle a?_{\pi p} \rangle t + (u \ \pi \oplus_{1-\pi} v) = (\langle a?_p \rangle t + u) \ \pi \oplus_{1-\pi} v \quad \text{(input-extract)}$$

Table 2. Axioms for Behavior Equivalence

Suppose now that $\alpha = \langle e, r \rangle \beta$. Let $s = \text{rt}(\langle a?_{\pi p} \rangle t + (u \ \pi \oplus_{1-\pi} v)) = \text{rt}(\langle a?_p \rangle t + u) \ \pi \oplus_{1-\pi} v$. In case $e \notin J \cup O$ (i.e. e is non-native), then applying Lemma 4 (1) and the induction hypothesis we have

$$\begin{aligned}
\mathcal{B}_{\langle a?_{\pi p} \rangle t + (u \ \pi \oplus_{1-\pi} v)}^O[\Phi](\langle e, r \rangle \beta) &= \mathcal{B}_{\langle a?_{\pi p} \rangle t + (u \ \pi \oplus_{1-\pi} v)}^O[\langle e, r + s \rangle^{-1} \Phi](\beta) \\
&= \mathcal{B}_{\langle a?_p \rangle t + u}^O \ \pi \oplus_{1-\pi} v[\langle e, r + s \rangle^{-1} \Phi](\beta) \\
&= \mathcal{B}_{\langle a?_p \rangle t + u}^O \ \pi \oplus_{1-\pi} v[\Phi](\langle e, r \rangle \beta).
\end{aligned}$$

It remains for us to consider the case $e \in J \cup O$. We compute, using Lemma 4:

$$\begin{aligned}
&\mathcal{B}_{\langle a?_{\pi p} \rangle t + (u \ \pi \oplus_{1-\pi} v)}^O[\Phi](\langle e, r \rangle \beta) \\
&= \mathcal{B}_{\langle a?_{\pi p} \rangle t}^O[\Phi](\langle e, r + s \rangle \beta) + \mathcal{B}_{u \ \pi \oplus_{1-\pi} v}^O[\Phi](\langle e, r + 0 \rangle \beta) \\
&= \pi \cdot \mathcal{B}_{\langle a?_p \rangle t}^O[\Phi](\langle e, r + s \rangle \beta) + \pi \cdot \mathcal{B}_u^O[\Phi](\langle e, r \rangle \beta) + (1 - \pi) \cdot \mathcal{B}_v^O[\Phi](\langle e, r \rangle \beta) \\
&= \pi \cdot \mathcal{B}_{\langle a?_p \rangle t + u}^O[\Phi](\langle e, r \rangle \beta) + (1 - \pi) \cdot \mathcal{B}_v^O[\Phi](\langle e, r \rangle \beta) \\
&= \mathcal{B}_{\langle a?_p \rangle t + u}^O \ \pi \oplus_{1-\pi} v[\Phi](\langle e, r \rangle \beta).
\end{aligned}$$

Note that the first and third steps crucially depend on the fact that the input-prefixed terms $\langle a?_{\pi p} \rangle t$ and $\langle a?_p \rangle t$ have aggregate rate 0, and

that $s = \text{rt}(\langle a?_{\pi p} \rangle t + (u \pi \oplus_{1-\pi} v)) = \text{rt}(u \pi \oplus_{1-\pi} v) = \text{rt}(u) = \text{rt}(v) = \text{rt}(\langle a?_p \rangle t + u) \pi \oplus_{1-\pi} v$.

4.4 Normal Forms

Let the notions *input normal form*, *output normal form*, and *normal form* be defined mutually recursively as follows:

- An *input normal form* is a well-typed term u that is either nil_I for some $I \neq \emptyset$, or else has the form: $\sum_{a \in I} \sum_{s \in R_a} \langle a?_{p_{a,s}} \rangle t_{a,s}$, where we require that:
 1. $I \neq \emptyset$.
 2. Each $t_{a,s}$ is a normal form, with $\text{rt}(t_{a,s}) = s$.
 3. For each $a \in I$ the set R_a is a nonempty finite subset of $(0, \infty)$.
 4. u is not an instance (up to permutation of sums) of the left-hand side of axiom (nil-fold).
- An *output normal form* is a well-typed term u that is either nil_\emptyset or else has the form: $\sum_{b \in O} \sum_{s \in R_b} \langle b!_{\sigma_{b,s} r} \rangle t_{b,s}$, where we require that:
 1. $O \neq \emptyset$.
 2. Each $t_{b,s}$ is a normal form, with $\text{rt}(t_{b,s}) = s$.
 3. For each $b \in O$ the set R_b is a nonempty finite subset of $(0, \infty)$.
 4. Each $\sigma_{b,s}$ satisfies $0 < \sigma_{b,s} \leq 1$ and $\sum_{b \in O} \sum_{s \in R_b} \sigma_{b,s} = 1$.
An output normal form is called *nontrivial* if it is not nil_\emptyset .
- A *normal form* is either an input normal form, an output normal form, or a sum $u + v$, where u is an input normal form and v is a nontrivial output normal form.

Lemma 8. *Any \parallel -free term t in $\text{Proc}(I/J \Rightarrow O)$ can be proved equivalent to a normal form using the axioms in Table 2.*

4.5 Completeness

Key to the completeness proof is Lemma 9 below, which shows how certain information about the structure of t can be extracted from its behavior. In case t is a normal form, this information is essentially the entire structure of t , except for the ordering of terms in sums. Suppose a type $I/J \Rightarrow O$ has been fixed and let $*$ be an arbitrarily chosen (non-native) action in $\text{Act} \setminus (J \cup O)$. Given $e \in \text{Act}$ and $r \geq 0$, let $\Xi_{e,r}$ be the observable defined as follows:

$$\Xi_{e,r}(\alpha) = \begin{cases} s, & \text{if } \alpha = \langle *, s \rangle, \\ 1, & \text{if } \alpha = \langle e, s \rangle \langle *, r \rangle, \\ 0, & \text{otherwise.} \end{cases}$$

We call such an observable a *probe*.

Lemma 9. *Suppose $t \in \text{Proc}(I/J \Rightarrow O)$. Then the probe $\Xi_{e,r}$ has the following properties:*

1. $\mathcal{B}_t^O[\Xi_{*,0}]((\ast, 0)) = \text{rt}(t)$.

2. For $e \in J \cup O$, $\mathcal{B}_t^O[\Xi_{e,r}]((e, 0)\langle \ast, 0 \rangle) = \sum_{\{u:\text{rt}(u)=r\}} \Delta_e^O(t, u)$.

Lemma 10. *Suppose t and t' are normal forms in $\text{Proc}(I/J \Rightarrow O)$. If $\mathcal{B}_t^O = \mathcal{B}_{t'}^O$, then t and t' are identical up to permutation of sums.*

Theorem 2. *The axioms in Table 2 are sound and complete for behavior equivalence.*

5 Conclusion

By comparing complete axiomatizations (and especially the normal forms arising in the completeness proofs), we have improved our understanding of the relationship between two notions of equivalence for processes with Markovian behavior. In contrast to the axiomatization of weighted bisimulation equivalence, the axiomatization of behavior equivalence exhibits differences in the role of input actions and output actions.

If we restrict to the output-only fragment of the language, then a complete axiomatization of behavior equivalence is given by axioms (choice-unit), (choice-comm), (choice-assoc), (comb-idemp), (comb-comm), (comb-assoc), (output-comb), (output-distr), and (interchange). This axiomatization may be compared to the axiomatization given in [Ber05] for the “Markovian trace equivalence” notion originally defined in [BC00]. In fact, each of the axioms for Markovian trace equivalence is sound for behavior equivalence, so (applying Bernardo’s completeness result) Markovian trace equivalent processes are also behavior equivalent.

Conversely, given a sequence of actions (*i.e.* a trace) $x \in \text{Act}^*$ and a time T in $[0, \infty)$, it is possible to define an observable $\Phi_{x,T}$ such that “the probability of performing an execution compatible with x in average time $\leq T$ ” is given by $\sum_{\alpha} \mathcal{B}_t^O[\Phi_{x,T}](\alpha)$, where α ranges over all rated traces that contain only actions in O . Thus, behavior equivalent output-only processes are also Markovian trace equivalent. So, one part of what we have achieved is to show that the introduction of the operator $\pi \oplus_{1-\pi}$ permits a finite axiomatization of Markovian trace equivalence, as opposed to the infinite axiom scheme given in [Ber05].

We have not yet succeeded in extending our results to include parallel composition. For weighted bisimulation equivalence there is an evident “expansion theorem” that permits parallel composition to be eliminated in favor of choice. For behavior equivalence, one might attempt a similar expansion for the parallel composition of two normal forms. One difficulty in doing this arises from the fact that behavior equivalence fails to be substitutive for parallel composition unless we restrict to input-stochastic terms. Thus we cannot employ various useful manipulations that move individual input-prefixed terms into and out of the scope of a parallel operator, as these do not preserve input-stochasticity, in general. Another subtlety is the following: if $t \in \text{Proc}(I/J \Rightarrow O)$ and $J' \cap (J \cup O) = \emptyset$,

then there is no way to eliminate parallel composition from a term of the form $t \circ \parallel_{O'} \text{nil}_J$. Such a term amounts to a kind of “input expansion” of t which, in the absence of a recursion operator, cannot be otherwise expressed. So in the absence of recursion there can be no expansion theorem that completely eliminates parallel composition. To attempt an axiomatization of recursion would first require an extension of the completeness results of the present paper to open terms. We leave these explorations as subjects for future research.

References

- [And99] Suzana Andova. Process algebra with probabilistic choice. In *ARTS '99: Proceedings of the 5th International AMAST Workshop on Formal Methods for Real-Time and Probabilistic Systems*, pages 111–129, London, UK, 1999. Springer-Verlag.
- [BC00] M. Bernardo and R. Cleaveland. A theory of testing for Markovian processes. In C. Palamidessi, editor, *CONCUR*, volume 1877 of *Lecture Notes in Computer Science*, pages 305–319, 2000.
- [BDG98] M. Bernardo, L. Donatiello, and R. Gorrieri. A formal approach to the integration of performance aspects in the modeling and analysis of concurrent systems. *Information and Computation*, 144(2):83–154, 1998.
- [Ber05] M. Bernardo. Markovian testing and trace equivalences exactly lump more than Markovian bisimilarity. In L. Aceto and A. D. Gordon, editors, *International Workshop on Algebraic Process Calculi: The First Twenty Five Years and Beyond (APC 25)*, Electronic Notes in Theoretical Computer Science. Springer-Verlag, 2005.
- [HH02] J.-P. Katoen H. Hermanns, U. Herzog. Process algebra for performance evaluation. *Theoretical Computer Science*, 274:43–97, 2002.
- [Hil96] J. Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.
- [HR94] H. Hermanns and M. Rettelbach. Syntax, semantics, equivalences, and axioms for MTIPP, 1994.
- [LS92] K. G. Larsen and A. Skou. Compositional verification of probabilistic processes. In *Proceedings of CONCUR '92*. Springer-Verlag Lecture Notes in Computer Science, 1992.
- [SCS03] E. W. Stark, R. Cleaveland, and S. A. Smolka. A process-algebraic language for probabilistic I/O automata. In Roberto M. Amadio and Denis Lugiez, editors, *CONCUR 2003 - Concurrency Theory, 14th International Conference, Marseille, France, September 3-5, 2003, Proceedings*, volume 2761 of *Lecture Notes in Computer Science*, pages 189–203. Springer-Verlag, 2003.
- [SCS06] E. W. Stark, R. Cleaveland, and S. A. Smolka. Probabilistic I/O automata: Theories of two equivalences. Full version available at <http://bsd7.cs.sunysb.edu/~stark/REPORTS/pioa-equivalences.pdf>, June 2006.
- [Sta03] E. W. Stark. On behaviour equivalence for probabilistic I/O automata and its relationship to probabilistic bisimulation. *Journal of Automata, Languages and Combinatorics*, 8(2):361–395, 2003.
- [WSS97] S.-H. Wu, S. A. Smolka, and E. W. Stark. Composition and behaviors of probabilistic I/O automata. *Theoretical Computer Science*, 176(1-2):1–38, 1997.