Compositional Analysis of Expected Delays in Networks of Probabilistic I/O Automata*

Eugene W. Stark[†] Scott A. Smolka[‡]

Department of Computer Science
SUNY at Stony Brook
Stony Brook, NY 11777-4400

Abstract

Probabilistic I/O automata *(PIOA) constitute a model for distributed or concurrent systems that incorporates a notion of probabilistic choice. The PIOA model provides a notion of* composition*, for constructing a PIOA for a composite system from a collection of PIOAs representing the components. We present a method for computing* completion probability *and* expected completion time *for PIOAs. Our method is* compositional*, in the sense that it can be applied to a system of PIOAs, one component at a time, without ever calculating the global state space of the system (*i.e. *the composite PIOA). The method is based on symbolic calculations with vectors and matrices of rational functions, and it draws upon a theory of* observables*, which are mappings from* delayed traces *to real numbers that generalize the classical "formal power series" from algebra and combinatorics. Central to the theory is a notion of* representation *for an observable, which generalizes the clasical notion "linear representation" for formal power series. As in the classical case, the representable observables coincide with an abstractly defined class of "rational" observables; this fact forms the foundation of our method.*

1 Introduction

In our previous paper [WSS97], we defined the class of *probabilistic I/O automata* (PIOA), which are a model for distributed or concurrent systems that incorporates a notion of probabilistic choice. The basic intuition underlying the model is the following: the time a PIOA spends in a state before performing its next action is described by an exponentially distributed random variable whose parameter (the so-called *delay parameter*) depends on the state. Under an independence assumption, a simple *composition rule* can be given for producing, given a collection of interacting PIOAs, a single "composite" PIOA representing the entire system.

We also showed how to associate with a PIOA a *probabilistic behavior map*, which in a sense represents the externally observable aspects of the behavior of the PIOA. We showed that behavior map semantics is compositional, in the sense that the behavior map associated with a composite PIOA is uniquely determined by the behavior maps associated with the components. We further showed that, for PIOAs satisfying a certain "delay restriction" concerning their internal actions, behavior map semantics is also fully abstract with respect to a behavioral equivalence based on a notion of probabilistic testing.

As a byproduct of the way of way probability is represented in the PIOA model, it is meaningful to consider certain aspects of timing for PIOA executions. In [Wu96], it is noted that the expected time for a PIOA to complete a specified finite sequence of actions (called a *trace*) can be extracted from the probabilistic behavior map associated with that automaton, and this idea was applied there to analyze some examples.

Certain limitations inherent in our previous work restricted its applicability as a method for analyzing expected completion times in a practical setting. A major problem was that our theory only supported "one trace at a time" analysis: given a PIOA A and a finite trace, the expected time for A to complete an execution having that trace could be determined, but the theory did not provide any useful method by which to specify an infinite set of traces and to determine the expected time for A to complete some execution having

*Effort sponsored by the Air Force Office of Scientific Research Air Force Materiel Command, USAF, under grant number F49620-96-1-0087. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

[†]E-mail address: `stark@cs.sunysb.edu`

[‡]E-mail address: `sas@cs.sunysb.edu`

one of the traces in that set. The latter problem, rather than the former, is the type of timing analysis that is more often encountered in practice. Another problem was that timing analysis could not be performed on a system of PIOAs "one component at a time"; essentially, a full description of the global state space system had to be constructed and the timing information extracted from that. Any "non-compositional" analysis method that requires the construction of the global state space of a system will in general only be able to handle very small systems, due to the exponential growth of the state space as the number of components increases.

In this paper, we present a new theory and associated analysis methods that overcome the limitations inherent in our previous work. One important part of our new theory is a revised definition of probabilistic behavior map which does not have the "trace at a time" limitation of our previous version. Our new definition makes use of a new notion of *delayed trace*, which generalizes to PIOAs the standard notion of the trace of an execution of an automaton, so that certain probabilistic scheduling information is represented along with the sequence of actions. An *observable* is defined to be a function from delayed traces to real numbers. The *behavior* of a PIOA is defined to be a *transformation of observables*; that is, a mapping from observables to observables. Our revised definition of PIOA behavior admits a much simpler compositionality result (Theorem 1) than the previous version. In particular, we show that the behavior of the composition of "compatible" probabilistic I/O automata is given by the ordinary function composition of the corresponding behaviors.

We show (Lemma 2) that information about *completion probability* and *expected completion time* for a PIOA A that is "closed" (*i.e.* has no input actions) can be obtained by applying its "empty alphabet behavior" $\mathcal{B}_\emptyset^A$ to appropriate observables. In particular, given a set T of finite action sequences, pairwise incomparable with respect to the prefix relation, one can define an observable Π_T, such that the value of $\mathcal{B}_\emptyset^A \Pi_T$ on a delayed trace (0) having no actions, is the probability of the set of executions of A whose delayed traces lie in the upward closure of T with respect to the prefix relation on delayed traces. We also define the "expected completion time" for A with respect to T to be the expected time for A to complete some execution having a delayed trace that "just reaches" the set T, and we show that, for a particular observable Ω_T, this time is given by the value of $\mathcal{B}_\emptyset^A \Omega_T$ on the delayed trace (0).

The above results lead to a method for computing the result of applying the behavior map for a system to a specific observable such as Π_T or Ω_T by working compositionally, "a component at a time," in such a way that the global state space is never constructed. This method is based on the realization that the observables Π_T and Ω_T can be represented in a certain way by a by a kind of automata, having states in a finite-dimensional vector space over the reals, that execute on delayed traces. We call such observables *representable*. We also show (Theorems 3 and 4) that the class of representable observables is closed under the application of PIOA behaviors, and that the result of applying a PIOA behavior to a representable observable can be effectively computed in terms of a construction on representations. Although this construction is a kind of "product construction," which produces an output representation whose size depends on the product of the size of the input representation and the number of states in the PIOA, we can mitigate the blow-up in size by applying a minimization algorithm to the result. We present a minimization algorithm (Theorem 5) that, given the representation of an observable as input, outputs a representation that in a sense has minimum size over all representations of the same observable.

Our theory of observables and their representations can be seen as a generalization of work by Carlyle and Paz [CP71], Schützenberger [Sch61a, Sch61b], and others (see [BR84] for references), on formal power series and linear representations. In particular, our "observables" generalize "formal power series," our "representations" generalize the "linear representations" for formal power series, and our "representable observables" generalize "recognizable series." We define a class of *rational observables*, which are those for which an associated space of *derivatives* is a finite-dimensional vector space, and we show (Theorem 2) that an observable is rational if and only if it is representable. This in a sense generalizes to observables a result of Carlyle and Paz [CP71], which equates the recognizable series with those whose "syntactic right ideal" has finite codimension. Our minimization algorithm for representations of observables corresponds to a result of Schützenberger [Sch61a, Sch61b] for formal power series. The novel aspects of our work are: (1) the introduction of "delayed traces" as a generalization of "words over a finite alphabet", and "observables" as a generalization of "formal power series"; (2) the recognition that "transformations of observables" yield a compositional semantics for PIOAs that is expressive enough to permit the treatment of expected termination time; (3) extension of the theory of "linear representations of formal power series" to a theory of "representable observables"; and (4) use of the theory of representable observables as a basis for deriving compositional algorithms for the analysis of PIOAs. Though closed PIOA's are examples of continuous-time semi-Markov processes [How71], and as such have a variety of well-developed analysis techniques applicable to them, we are not aware of such techniques that do not have as a prerequisite the construction of a global system description such as a transition matrix or flowgraph.

In other related work, Campos et al. in [CCM97] present BDD-based algorithms that determine the exact bounds on the delay between two specified events and the number of occurrences of another event in all such intervals. Segala et al. [LSS94, PS95] have developed a method for the analysis of the expected time complexity of randomized distributed algorithms. The method consists of proving auxiliary probabilistic time bound statements of the form $U \text{---}\{t,p\} \rightarrow U'$, which mean that whenever the algorithm begins in a state in a set U, it will reach a state in set U' within time t with probability at least p. Finally, a number of "stochastically timed" process algebras and Petri net formalisms have been proposed for the performance analysis of concurrent systems, including [MBC84, GHR93, Hil96, Pri96, BDG98]. In the case of process algebra, these approaches are sometimes referred to as "compositional" in the sense that a composite stochastic system can be specified algebraically in terms of its components.

The remainder of this paper is organized as follows: Section 2 is devoted to the basic definitions and theory of PIOAs and probabilistic behavior maps. Section 3 treats representable observables. Section 4 presents our main results. Finally, Section 5 considers briefly a simple example of the use of the techniques. Due to space limitations, we have omitted all proofs, replacing them with sketches in the most important cases. Full versions of all definitions and proofs can be found in [SS97].

2 Probabilistic I/O Automata and Their Behaviors

2.1 Probabilistic I/O Automata

In this section, we recall the basic definitions from [WSS97]. We give here simplified versions of the definitions, which are equivalent to those of [WSS97] in the case of *finite* PIOAs, which are all that we consider in the present paper. The reader should refer to [WSS97] and [SS97] for full details and further discussion.

A *finite probabilistic I/O automaton* is a tuple $A = (Q, q^{\mathrm{I}}, E, \Delta, \mu, \delta)$, where

- Q is a finite set of *states*;
- $q^{\mathrm{I}} \in Q$ is a distinguished *start state*;
- E is a finite set of *actions*, partitioned into disjoint sets of *input*, *output*, and *internal* actions, which are denoted by E^{in}, E^{out}, and E^{int}, respectively, with the actions in $E^{\mathrm{loc}} = E^{\mathrm{out}} \cup E^{\mathrm{int}}$ called *locally controlled*;
- $\Delta \subseteq Q \times E \times Q$ is the *transition relation*, which satisfies the following *input-enabledness* condition: for any state $q \in Q$ and input action $e \in E^{\mathrm{in}}$, there exists a state $r \in Q$ such that $(q, e, r) \in \Delta$.
- $\mu : (Q \times E \times Q) \rightarrow [0, 1]$ is the *transition probability* function, which is required to satisfy the following stochasticity conditions:
  1. $\mu(q, e, r) > 0$ iff $(q, e, r) \in \Delta$.
  2. $\sum_{r \in Q} \mu(q, e, r) = 1$, for all $q \in Q$ and all $e \in E^{\mathrm{in}}$.
  3. For all $q \in Q$, if there exist $e \in E^{\mathrm{loc}}$ and $r \in Q$ such that $(q, e, r) \in \Delta$, then $\sum_{r \in Q} \sum_{e \in E^{\mathrm{loc}}} \mu(q, e, r) = 1$,
- $\delta : Q \rightarrow [0, \infty)$ is the *state delay* function, which is required to satisfy the following condition: for all $q \in Q$, we have $\delta(q) > 0$ if and only if there exist $e \in E^{\mathrm{loc}}$ and $r \in Q$ such that $(q, e, r) \in \Delta$.

As discussed in [WSS97], the definitions of μ and δ above reflect the intuition we wish to capture concerning the execution of a system of PIOAs. Upon arrival in a state q, a PIOA chooses randomly the length of time it will spend in that state before executing its next "locally controlled" transition. The random choice is made, independently of the other PIOAs in the system, according to an exponential holding time distribution whose mean is the reciprocal $1/\delta(q)$ of the delay parameter $\delta(q)$ associated with that state.

Our definition of PIOA is a "exponential semi-Markov" definition, in which the "holding time" in state q before the next locally controlled action occurs is described by a random variable having an exponential distribution with mean $1/\delta(q)$. Once the holding time in state q has expired, the action e and successor state r is chosen randomly with probability $\mu(q, e, r)$. As an alternative to the exponential semi-Markov description, we could have chosen an "exponential Markov description", in which associated with each locally controlled transition (q, e, r) is a "transition rate" $\rho(q, e, r)$ that describes the "probability flux" from state q to state r via the action e. For e an input action, we would still need the probabilities $\mu(q, e, r)$. It is not difficult to see that the two forms of description are entirely equivalent, A *finite execution fragment* for a probabilistic I/O automaton A is an alternating sequence of states and actions of the form

$$q_0 \xrightarrow{e_0} q_1 \xrightarrow{e_1} \ldots \xrightarrow{e_{n-1}} q_n,$$

such that for each k with $0 \leq k < n$, either $e_k \in E$ and $(q_k, e_k, q_{k+1}) \in \Delta$, or else $e_k \notin E$ and $q_{k+1} = q_k$. An execution fragment with $q_0 = q^{\mathrm{I}}$ (the distinguished start state) is called an *execution*. If σ denotes an execution fragment, then we will use $\sigma(k)$ to denote the state q_k, for $0 \leq k \leq n$, and we will use $\sigma(k, k+1)$ to denote the action e_k, for $0 \leq k < n$. The number n is called the *length* of σ, and we denote it by $|\sigma|$. We use the term *trace* to refer to a sequence of actions. The *trace of* σ, denoted $\mathrm{tr}(\sigma)$, is the sequence of actions $e_0 e_1 \ldots e_{n-1}$ appearing in σ.

In [WSS97], we showed how a closed PIOA A (one with no input actions) induces a probability space over the set of all its executions. The probability measure pr_A is derived in a natural way from the mapping that assigns to each execution $\sigma = q_0 \xrightarrow{e_0} q_1 \xrightarrow{e_1} \ldots \xrightarrow{e_{n-1}} q_n$ the quantity $\mathrm{p}_A(\sigma) = \prod_{k=0}^{n-1} \mu(q_k, e_k, q_{k+1})$, where by convention, we take $\mu(q, e, q) = 1$ if $e \notin E$. In this paper, we shall also be interested in the related function $\mathrm{w}_A(\sigma)$ on finite executions, defined by $\mathrm{w}_A(\sigma) = \mathrm{p}_A(\sigma) \prod_{\{k : e_k \in E_A^{\mathrm{loc}}\}} \delta_A(q_k)$. Although the definition of w_A may at first seem somewhat *ad hoc*, it turns out that w_A behaves in a more convenient fashion than $\mathrm{p}_A(\sigma)$ when considering the composition of PIOAs. In particular, w_A has the useful property stated in Lemma 1 below, which makes possible the statement of a compositionality law that does not have the ugly normalization factor $h(\mathbf{d}^A, \mathbf{d}^B)$ present in our previous paper [WSS97].

A finite collection $\{A_i : i \in I\}$ of probabilistic I/O automata, where $A_i = (Q_i, q_i^{\mathrm{I}}, E_i, \Delta_i, \mu_i, \delta_i)$, is called *compatible* if for all $i, j \in I$, $i \neq j$, we have $E_i^{\mathrm{out}} \cap E_j^{\mathrm{out}} = \emptyset$ and $E_i^{\mathrm{int}} \cap E_j = \emptyset$. The *composition* $\|_{i \in I} A_i$ of a finite compatible collection is a probabilistic I/O automaton $(Q, q^{\mathrm{I}}, E, \Delta, \mu, \delta)$, defined as follows:

- $Q = \prod_{i \in I} Q_i$.
- $q^{\mathrm{I}} = \langle q_i^{\mathrm{I}} : i \in I \rangle$.
- $E = \bigcup_{i \in I} E_i$, where $E^{\mathrm{out}} = \bigcup_{i \in I} E_i^{\mathrm{out}}$, $E^{\mathrm{int}} = \bigcup_{i \in I} E_i^{\mathrm{int}}$, and $E^{\mathrm{in}} = (\bigcup_{i \in I} E_i^{\mathrm{in}}) \setminus E^{\mathrm{out}}$.
- Δ is the set of all $(\langle q_i : i \in I \rangle, e, \langle r_i : i \in I \rangle)$ such that for all $i \in I$, if $e \in E_i$, then $(q_i, e, r_i) \in \Delta_i$, otherwise $r_i = q_i$.
- $\delta(\langle q_i : i \in I \rangle) = \sum_{i \in I} \delta_i(q_i)$.
- If $e \in E^{\mathrm{in}}$, then

  $$\mu(\langle q_i : i \in I \rangle, e, \langle r_i : i \in I \rangle) = \prod_{i \in I} \mu_i(q_i, e, r_i).$$

  If $e \in E_k^{\mathrm{loc}}$ for some k, then

  $$\mu(\langle q_i : i \in I \rangle, e, \langle r_i : i \in I \rangle) = \frac{\delta_k(q_k)}{\sum_{i \in I} \delta_i(q_i)} \prod_{i \in I} \mu_i(q_i, e, r_i).$$

We use the notation $A \| B$ to denote the composition $\|\{A, B\}$ of a compatible 2-element set of PIOAs.

The definitions of μ and δ for the composition of PIOAs expresses the intuitive idea that the various component PIOAs are in a race to see which of them will execute the next locally controlled action. This competition will be won by the component that has chosen the smallest holding time in its respective state, and the probability that any given component will win the competition is given by the ratio of the local delay parameter for that component over the sum of the local delay parameters for all components. The time the system remains in a particular global state before executing the next locally controlled action is the minimum of the times that each component spends in its respective local state. This time is governed by an exponential distribution, whose parameter is the sum of the parameters of the distributions for each of the components. Note that the definition of composition of PIOAs involves synchronization (via shared input and output actions) between component automata. It is precisely this type of synchronization or interaction between subsystems that tends to destroy so-called "product form" properties that have been found very useful in finding analytical solutions to classical queueing network problems [Dij93].

2.2 Delayed Traces, Observables, and Behaviors

Let E be a set of actions. A (finite) *delayed trace* α *over* E consists of an alternating sequence of the form:

$$d_0 \xrightarrow{e_0} d_1 \xrightarrow{e_1} \ldots \xrightarrow{e_{n-1}} d_n,$$

where the d_k are nonnegative real numbers and the the e_k are actions in E. The sequence $e_0, e_1, \ldots e_{n-1}$ is called the *trace of* α, and we denote it by $\mathrm{tr}(\alpha)$. The sequence $d_0, d_1, \ldots, d_n$ is called the *sequence of delay parameters* of α. We use the notation $\alpha(k)$ to denote d_k, and the notation $\alpha(k, k+1)$ to denote e_k. The number n is called the *length* of α, and we denote it by $|\alpha|$.

We use $\mathrm{DTraces}(E)$ to denote the set of all delayed traces over E. We also use the notation $(d)_E$, or just

(d), when E is clear from the context, to denote the *empty delayed trace* in DTraces(E), consisting of the single delay parameter d and no actions.

Suppose $\alpha \in$ DTraces(E). If $E \subseteq E'$, then a delayed trace $\alpha' \in$ DTraces(E') is a *refinement* of α, and we write $\alpha' \rhd \alpha$, if α' can be obtained from α by inserting into α a finite number of actions from $E' \setminus E$, where the delay parameters at the newly created positions in α' are obtained by "stuttering" (repeating the previous value from α). Figure 1 (a) depicts graphically the refinement relationship between α' and α.

Suppose A is a PIOA. If α is a delayed trace over E, then an execution σ of A is *conformant with* α, and we write $\sigma \propto \alpha$, if the sequence of actions occurring in σ contains the sequence of actions of α as a subsequence, in such a way that any actions in σ that do not correspond to actions of α are elements of the set $E_A \setminus E$. Note, in particular, that conformance does not require any relationship between the delay parameters of the states in σ and the delay parameters in α. Figure 1 (b) depicts graphically the conformance relationship between σ and α.

Suppose $\alpha \in$ DTraces(E) and $\sigma \propto \alpha$. Then the *combination* of σ and α is the delayed trace α'' characterized uniquely by the following conditions: $|\alpha''| = |\sigma|$, $\sigma \propto \alpha''$, and for each k with $0 \leq k \leq |\alpha''|$, the delay parameter $\alpha''(k)$ is the sum of $\delta(\sigma(k))$ and $\alpha(j_k)$, where j_k is the position in α corresponding under the refinement relationship to the position k in α''. We write $\sigma \oplus \alpha$ to denote the $\alpha'' \in$ DTraces$(E \cup E_A)$ that is the combination of σ and α. Figure 1 (c) depicts graphically the result of combining σ and α.

An *observable* over a set of actions E is a mapping $\Phi :$ DTraces$(E) \rightarrow \mathcal{R}$, where $\mathcal{R}$ denotes the set of real numbers. If A is a PIOA and E is a set of actions, then the *E-behavior* of A is the *transformation of observables*:

$$\mathcal{B}^A_E : (\text{DTraces}(E \cup E_A) \rightarrow \mathcal{R}) \rightarrow (\text{DTraces}(E) \rightarrow \mathcal{R})$$

defined by:

$$\mathcal{B}^A_E \Phi \alpha = \sum_{\sigma \propto \alpha} \Phi(\sigma \oplus \alpha) \mathrm{w}_A(\sigma).$$

In general, $\mathcal{B}^A_E \Phi \alpha$ will not be defined for all Φ and α, because the defining summation above need not converge. However, for observables Φ that "approach 0 quickly enough," $\mathcal{B}^A_E \Phi \alpha$ will be defined.

The concepts defined above have the following intuitive significances. A delayed trace denotes an equivalence class of executions of a PIOA, such that all executions in the class contain the same actions in the same order, and such that the sequences of delay parameters associated with the states traversed are the same (though the state sequences themselves may differ). An observable should be thought of as an abstract summary of the behavior of a part of a system, insofar as it pertains to a particular quantity of interest we are trying to measure. We will show in Section 3.1 how to define observables corresponding to two quantities of interest: (1) the probability of performing an execution having a delayed trace that lies in a specified set, and (2) the expected time to complete an execution having a given delayed trace. The E-behavior $\mathcal{B}^A_E$ of a PIOA A constitutes an abstract description of A as a kind of transducer, which takes as its argument an observable Φ summarizing, with respect to a particular quantity of interest, the behavior of a part of a system that is to be composed with A, and produces as its result an observable $\mathcal{B}^A_E \Phi$ that summarizes, with respect to the same quantity of interest, the behavior of the given part of the system together with A.

The definition of the E-behavior of a PIOA has an operational reading: "Given an observable Φ (summarizing a part of a system) and a delayed trace α (representing constraints on the set of executions to be examined), enumerate all executions σ of PIOA A that extend (*i.e.* are conformant with) α, in the sense of containing all the actions of α, possibly interleaved with some additional actions of A. For each such execution σ, construct a delayed trace $\sigma \oplus \alpha$ representing the combination of the constraints α plus the new constraints imposed by σ, apply Φ to $\sigma \oplus \alpha$, weight the resulting quantity by $\mathrm{w}_A(\sigma)$, and sum." It is perhaps useful to think of the function $\mathcal{B}^A_E$ as "almost an expectation operator," which produces a kind of expected value for the observable Φ supplied as its argument. The operator $\mathcal{B}^A_E$ fails to be a true expectation operator, however, because the quantities $\mathrm{w}_A(\sigma)$ do not constitute a probability distribution.

2.3 Compositionality

Although the definition of the behavior $\mathcal{B}^A_E$ is quite technical, the justification for it is that it satisfies a very simple compositionality result, which shows how the behavior $\mathcal{B}^{A\|B}_E$ for a composite PIOA $A\|B$ can be derived from the component behaviors $\mathcal{B}^A_E$ and $\mathcal{B}^B_{E \cup E_A}$.

Theorem 1 *Suppose A and B are compatible PIOAs, and E is a set of actions. Then $\mathcal{B}^{A\|B}_E = \mathcal{B}^A_E \circ \mathcal{B}^B_{E \cup E_A}$.*

The proof of Theorem 1 depends on Lemma 1 below, which is similar in nature to Proposition 2.1 in [WSS97]. With it, the proof of Theorem 1 is simply a matter of expanding the expression $\mathcal{B}^A_E \circ \mathcal{B}^B_{E \cup E_A}$ using the definition of behavior, then using the Lemma to combine the double summation into a single summation.

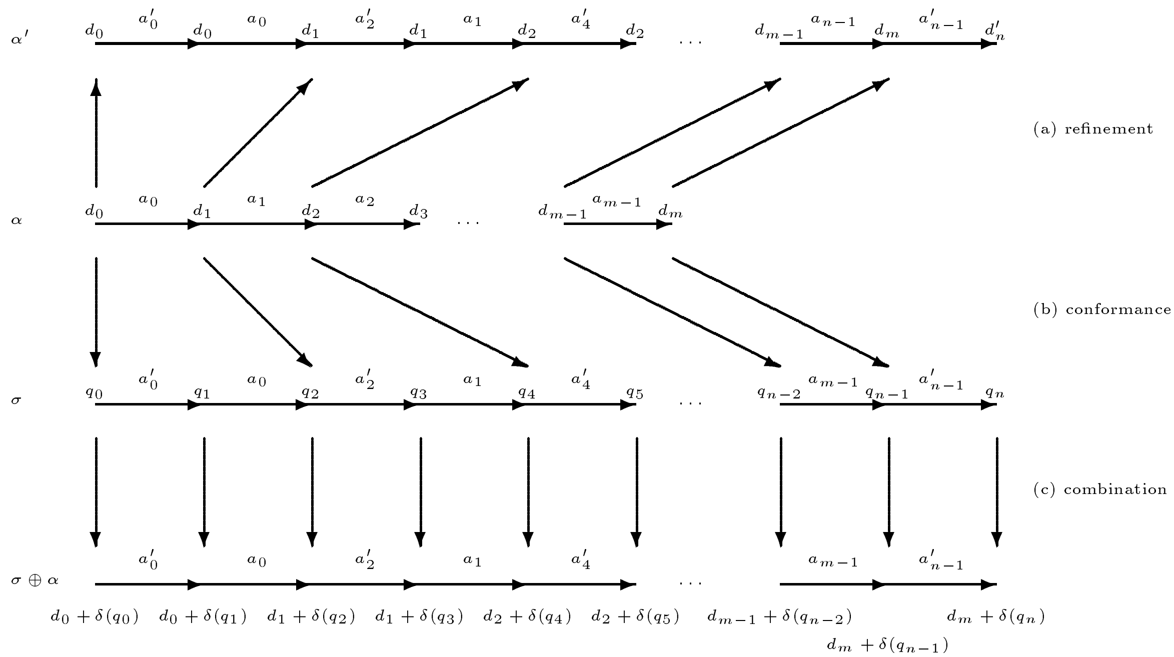

Figure 1: Refinement, Conformance, and Combination

Lemma 1 *Suppose A and B are compatible PIOAs. Then, given a delayed trace α, the set of all executions σ of $A\|B$ such that $\sigma \propto \alpha$, is in bijective correspondence with the set of all pairs of executions (σ^A, σ^B), where σ^A is an execution of A such that $\sigma^A \propto \alpha$, and σ^B is an execution of B such that $\sigma^B \propto (\sigma^A \oplus \alpha)$. Moreover, whenever σ corresponds under the bijection to the pair (σ^A, σ^B) we have:*

$$\sigma \oplus \alpha = \sigma^B \oplus (\sigma^A \oplus \alpha)$$

and

$$\mathrm{w}_{A\|B}(\sigma) = \mathrm{w}_A(\sigma^A)\, \mathrm{w}_B(\sigma^B).$$

2.4 Expected Completion Time

We are interested in calculating the expected time taken for a PIOA A to complete a finite execution having an action sequence in a specified set, which in general will be infinite. To avoid ambiguity surrounding executions that "complete" multiple times in the sense of having more than one prefix with an action sequence lying in the specified set, we restrict our attention to sets of action sequences that are pairwise incomparable with respect to the prefix relation. We call such a set a *target set*. If T is a target set, then we write $T\uparrow$ to denote the upward-closure of T with respect to the prefix relation.

If A is a closed PIOA, and T is a target set, then the *completion probability* $\mathrm{pr_c}(A,T)$ for A with respect to T is the quantity:

$$\mathrm{pr_c}(A,T) = \mathrm{pr}_A\{\sigma : \mathrm{tr}(\sigma) \in T\uparrow\}.$$

If $\mathrm{pr_c}(A,T) = 1$, then the *expected completion time* $\exp_c(A,T)$ for A with respect to T is the quantity:

$$\exp_c(A,T) = \sum_{\mathrm{tr}(\sigma)\,\in\, T} \left(\sum_{k=0}^{|\sigma|-1} \frac{1}{\delta_A(\sigma(k))} \right) \mathrm{p}_A(\sigma).$$

To understand the above definition, recall that the mean of an exponential distribution with parameter λ is $1/\lambda$, so that the quantity $1/\delta_A(\sigma(k))$ is the expected holding time for A in state $\sigma(k)$.

Suppose $T \subseteq E^*$ is a target set. The *characteristic observable* of T is the map $\chi_T : \mathrm{DTraces}(E) \to \mathcal{R}$ defined by: $\chi_T(\alpha) = 1$ if $\mathrm{tr}(\alpha) \in T$, and $\chi_T(\alpha) = 0$ otherwise. Two other observables will be of interest to us. The *completion probability* observable $\Pi : \mathrm{DTraces}(E) \to \mathcal{R}$ and the *completion time* observable $\Omega : \mathrm{DTraces}(E) \to \mathcal{R}$ are defined by

$$\Pi(\alpha) = \prod_{k=0}^{|\alpha|-1} \frac{1}{\alpha(k)}.$$

$$\Omega(\alpha) = \left(\sum_{k=0}^{|\alpha|-1} \frac{1}{\alpha(k)} \right) \left(\prod_{k=0}^{|\alpha|-1} \frac{1}{\alpha(k)} \right).$$

If T is a target set, then we define:

$$\Pi_T(\alpha) = \Pi(\alpha)\chi_T(\alpha) \qquad \Omega_T(\alpha) = \Omega(\alpha)\chi_T(\alpha).$$

The multiplicative factor of $\prod_{k=0}^{|\alpha|-1} \frac{1}{\alpha(k)}$ in the definition of Π and Ω above serves to make these observables "go to 0 quickly enough" to cancel out the blowup effect introduced by the weights $\mathrm{w}_A(\sigma)$ in the definition of $\mathcal{B}_E^A$. If A is a closed PIOA (*i.e.* has no input actions), then the cancellation is complete, so that applying the behavior $\mathcal{B}_E^A$ of a PIOA A to Π or Ω amounts to taking the expectation of the functions 1 and $\sum_{k=0}^{|\alpha|-1} \frac{1}{\alpha(k)}$, respectively. The following result states this formally.

Lemma 2 *Suppose A is a closed PIOA, and T is a target set. Then*

$$\mathrm{pr}_{\mathrm{c}}(A,T) = \mathcal{B}_\emptyset^A \Pi_T\ (0),$$

$$\mathrm{exp}_{\mathrm{c}}(A,T) = \mathcal{B}_\emptyset^A \Omega_T\ (0),$$

where (0) denotes the delayed trace with no actions and zero as its sole delay parameter.

If A is not closed, then there is a subtle effect here that is difficult to understand unless one carefully works through all the proofs. The cancellation of factors is not entirely complete, and the residue that is left is exactly what is needed to get the appropriate normalization factor that correctly accounts for the probability of a particular scheduling of locally controlled actions between A and its environment. To achieve the automatic introduction of this normalization factor is the reason why we defined the strange-looking quantities $\mathrm{w}_A(\sigma)$ and used them in the definition of $\mathcal{B}_E^A$.

3 Representable Observables

In this section, we develop the theory of "representable observables," to prepare the way for showing that this theory, together with that of the previous section, can be used to obtain a compositional method for computing expected completion time.

Let $\mathrm{Obs}(E)$ denote the set of all observables $\Phi : \mathrm{DTraces}(E) \to \mathcal{R}$. Then $\mathrm{Obs}(E)$ is a vector space under the usual pointwise addition and multiplication. Let $\mathrm{Rat}(x)$ denote the set of all real-valued rational functions of a single real parameter x. For n a nonnegative integer, an *n-dimensional representation* of an observable $\Phi \in \mathrm{Obs}(E)$ consists of

- An n-dimensional row vector C with entries in $\mathcal{R}$,
- An n-dimensional column vector $D(x)$ with entries in $\mathrm{Rat}(x)$,
- For each $a \in E$, an $n \times n$ matrix $M_a(x)$, with entries in $\mathrm{Rat}(x)$,

such that for all delayed traces $\alpha \in \mathrm{DTraces}(E)$, the quantity $\Phi(\alpha)$ is given by the formula:

$$\Phi(\alpha) = C \left(\prod_{k=0}^{|\alpha|-1} M_{\alpha(k,k+1)}(\alpha(k)) \right) D(\alpha(|\alpha|)),$$

An observable $\Phi \in \mathrm{Obs}(E)$ is called *representable* if there exists an n-dimensional representation of Φ, for some n.

A representation is essentially a kind of automaton that computes a function on delayed traces (*i.e.* an observable). The states of the automaton are n-dimensional row vectors of real numbers, with the vector C serving as the initial state. If the automaton is in state X, and the next portion of the input is $d\xrightarrow{a}$, then the automaton multiplies the current state vector by the matrix $M_a(d)$, and advances the input pointer. Upon reaching the end of the input, if the current state is X' and the single remaining delay parameter is d, then the row vector X' is multiplied by the column vector $D(d)$, to obtain a scalar, which becomes the output produced by the automaton.

Suppose $\Phi : \mathrm{DTraces}(E) \to \mathcal{R}$ is an observable. If $d \in \mathcal{R}$ and $a \in E$, then the *derivative* of Φ by d and a is the observable $\Phi_{d\xrightarrow{a}}$ defined by:

$$\Phi_{d\xrightarrow{a}}(\alpha) = \Phi(d\xrightarrow{a}\alpha),$$

where if α is the delayed trace:

$$d_0 \xrightarrow{e_0} d_1 \xrightarrow{e_1} \dots \xrightarrow{e_{n-1}} d_n,$$

then $d\xrightarrow{a}\alpha$ denotes the delayed trace:

$$d \xrightarrow{a} d_0 \xrightarrow{e_0} d_1 \xrightarrow{e_1} \dots \xrightarrow{e_{n-1}} d_n.$$

It is not difficult to verify that for all $d \in \mathcal{R}$ and $a \in E$, the mapping taking $\Phi \in \mathrm{Obs}(E)$ to its derivative $\Phi_{d\xrightarrow{a}} \in \mathrm{Obs}(E)$ is a linear transformation on $\mathrm{Obs}(E)$.

If S is an arbitrary subset of $\mathrm{Obs}(E)$, then define

$$\mathcal{D}S = \{\Phi_{d\xrightarrow{a}} : \Phi \in S, d \in \mathcal{R}, a \in E\},$$

and let $\mathcal{D}^*S$ denote the least subspace of $\mathrm{Obs}(E)$ containing S and satisfying $\mathcal{D}(\mathcal{D}^*S) \subseteq \mathcal{D}^*S$.

Define an observable $\Phi \in \mathrm{Obs}(E)$ to be *rational* if the following three conditions hold:

1. The space $\mathcal{D}^*\Phi$ is a finite-dimensional subspace of $\mathrm{Obs}(E)$.
2. For all $\Psi \in \mathcal{D}^*\Phi$, the quantity $\Psi(d)$ (the value of Ψ on the delayed trace of length zero with single delay parameter d) is a rational function of d.
3. For all $\Psi \in \mathcal{D}^*\Phi$, all $a \in E$, and all linear maps $L : \mathcal{D}^*\Phi \to \mathcal{R}$, the quantity $\Psi_{d\xrightarrow{a}}L$ is a rational function of d (note that we denote the application of a linear transformation by writing it to the right of its argument).

Define the *dimension* of a rational observable Φ to be the dimension of the space $\mathcal{D}^*\Phi$.

Theorem 2 *An observable $\Phi \in \mathrm{Obs}(E)$ is rational if and only if it is representable. Moreover, if an observable Φ is representable, then it has a representation whose dimension is equal to the dimension of Φ, and this dimension is the minimum possible among representations of Φ.*

The coincidence of representable and rational observables, as expressed by Theorem 2, is a key fact in the proofs of our results.

3.1 Examples of Representable Observables

In this section we show that certain observables of interest are representable.

Lemma 3 *The completion probability observable Π has the 1-dimensional representation*

$$C = (1) \qquad D(x) = (1)$$

$$M_a(x) = (1/x).$$

The completion time observable Ω has the 2-dimensional representation

$$C = \begin{pmatrix} 0 & 1 \end{pmatrix} \qquad D(x) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$M_a(x) = \begin{pmatrix} 1/x & 0 \\ 1/x^2 & 1/x \end{pmatrix}.$$

Lemma 4 *Suppose T is a target set which is also a regular subset of E^*. Then the observables Π_T and Ω_T are representable, with representations that can be effectively computed from the description of a deterministic finite automaton that recognizes T.*

4 Representable Observables and PIOA Behaviors

This section presents our main results. Theorem 3 says that the result of applying a PIOA behavior to a representable observable can be computed explicitly in terms of representations. Theorem 4 tells us how to use representations to "restrict the action set" of a behavior. Finally, Theorem 5 gives an algorithm that, given a representation for an observable, computes a representation of minimal dimension for that same observable.

Theorem 3 *Suppose A is a PIOA. If Φ is a representable observable in $\mathrm{Obs}(E)$, where $E_A \subseteq E$, then $\mathcal{B}^A_E\Phi$ is also a representable observable in $\mathrm{Obs}(E)$. Moreover, a representation of $\mathcal{B}^A_E\Phi$ can be effectively computed from a representation of Φ.*

Proof Sketch – Suppose $\Phi \in \mathrm{Obs}(E)$ is a representable observable, where $E_A \subseteq E$, and let

$$(C, D(x), \{M_a(x) : a \in E\})$$

be an n-dimensional representation of Φ. Suppose the PIOA A has m states. Let $q_1, q_2, \ldots, q_m$ be an enumeration of the states of A, with q_1 the distinguished start state. We construct an mn-dimensional representation $(C', D'(x), \{M'_a(x) : a \in E\})$ of $\mathcal{B}^A_E\Phi$, where C', $D'(x)$, and $M'_a(x)$ are given in n-dimensional block form as follows:

$$C' = \begin{pmatrix} C & 0 & \ldots & 0 \end{pmatrix}$$

$$D'(x) = \begin{pmatrix} D(x + \delta_A(q_1)) \\ D(x + \delta_A(q_2)) \\ \ldots \\ D(x + \delta_A(q_m)) \end{pmatrix}$$

$$(M'_a)_{ij}(x)$$
$$= \begin{cases} \mu_A(q_i, a, q_j) M_a(x + \delta_A(q_i)) \delta_A(q_i), & \text{if } a \in E_A^{\mathrm{loc}}, \\ \mu_A(q_i, a, q_j) M_a(x + \delta_A(q_i)), & \text{otherwise.} \end{cases}$$

The following is the basic correctness property for the above representation.

Claim: *For all delayed traces α in $\mathrm{DTraces}(E)$ of the form:*

$$d_0 \xrightarrow{a_0} d_1 \xrightarrow{a_1} \ldots \xrightarrow{a_{l-1}} d_l,$$

the jth n-dimensional block of the row vector:

$$C' \left(\prod_{k=0}^{l-1} M'_{\alpha(k,k+1)}(\alpha(k)) \right)$$

is equal to the following sum:

$$\sum_{\sigma \in \Xi_A(\alpha, q_j)} \mathrm{w}_A(\sigma)\, C \prod_{k=0}^{l-1} M_{\sigma(k,k+1)}(\alpha(k) + \delta_A(\sigma(k)))$$

where $\Xi_A(\alpha, q_j)$ denotes the set of all executions of A of the form:

$$r_0 \xrightarrow{a_0} r_1 \xrightarrow{a_1} \ldots \xrightarrow{a_{l-1}} r_l,$$

with $r_0 = q_1$ and $r_l = q_j$. ∎

We now show how to extend the previous result to the case of $\mathcal{B}^A_E$, where we do not necessarily have $E_A \subseteq E$.

If $E \subseteq E'$, then define the map $[\text{ - }]_E : \mathrm{Obs}(E') \to \mathrm{Obs}(E)$ by: $[\Psi]_E\alpha = \sum_{\alpha' \rhd \alpha} \Psi(\alpha')$. Note that the sum on the right need not converge, in general, so that $[\Psi]_E$ will be defined only for certain $\Psi \in \mathrm{Obs}(E')$.

The following result states that the E-behavior of A is already determined by the $(E \cup E_A)$-behavior of A.

Lemma 5 *Suppose A is a PIOA. Then for all sets of actions E, and for all observables $\Phi \in \mathrm{Obs}(E \cup E_A)$, we have: $\mathcal{B}^A_E \Phi = [\mathcal{B}^A_{E \cup E_A} \Phi]_E$.*

In view of Lemma 5, the map $[\,\text{-}\,]_E$ allows us to reduce the problem of computing a representation of $\mathcal{B}^A_E \Phi$ to that of computing, given a representation of $\mathcal{B}^A_{E \cup E_A} \Phi$, a representation of the observable $[\mathcal{B}^A_{E \cup E_A} \Phi]_E$. Theorem 4 below gives a method for computing $[\,\text{-}\,]_E$ on representations.

Theorem 4 *Suppose $(C', D', \{M'_{a'} : a' \in E'\})$ is a representation of an observable $\Phi' \in \mathrm{Obs}(E')$, and suppose $E \subseteq E'$. Suppose further that $[\,\text{-}\,]_E$ is well defined on $\mathcal{D}^* \Phi'$ and that that the power series:*

$$I + \hat{M}(x) + \hat{M}^2(x) + \dots$$

converges (componentwise) for all nonnegative $x \in \mathcal{R}$, where we define

$$\hat{M}(x) = \sum_{a' \in E' \setminus E} M'_{a'}(x).$$

Then the matrix $I - \hat{M}(x)$ is nonsingular for all nonnegative $x \in \mathcal{R}$, and an n-dimensional representation of $[\Phi']_E$ is given by

$$(C', \;\; \hat{M}(x)^* D'(x), \;\; \{\hat{M}(x)^* M'_a(x) : a \in E\}),$$

where $\hat{M}(x)^*$ abbreviates $(I - \hat{M}(x))^{-1}$.

The proof of Theorem 4 makes use of the following lemma, which gives recursive rules for computing the derivatives of an observable of the form $[\Psi]_E$ and for evaluating an observable of the form $[\Psi]_E$ on a delayed trace (d) of length 0.

Lemma 6 *Suppose $E \subseteq E'$. Suppose further that S is a linear subspace of $\mathrm{Obs}(E')$ such that:*

1. *$\mathcal{D} S \subseteq S$.*
2. *$[\Psi]_E$ is defined for all $\Psi \in S$.*

Then the following relations are satisfied for all $\Psi \in S$, all $d \in \mathcal{R}$, and all $a \in E$:

$$([\Psi]_E)_{d \xrightarrow{a}} = [\Psi_{d \xrightarrow{a}}]_E + \sum_{a' \in E' \setminus E} ([\Psi_{d \xrightarrow{a'}}]_E)_{d \xrightarrow{a}}$$

$$[\Psi]_E(d) = \Psi(d) + \sum_{a' \in E' \setminus E} [\Psi_{d \xrightarrow{a'}}]_E(d).$$

The following gives sufficient conditions for the technical hypotheses of the previous theorem to hold.

Lemma 7 *Suppose $(C', D'(x), \{M'_{a'} : a' \in E'\})$ is a representation of $\Phi' \in \mathrm{Obs}(E')$, and suppose $E \subseteq E'$. Let $\hat{M}(x) = \sum_{a' \in E' \setminus E} M'_{a'}(x)$. Suppose*

1. *$M_{a'}(x) \geq 0$ and $D'(x) \geq 0$ componentwise for all $a' \in E'$ and all nonnegative $x \in \mathcal{R}$.*
2. *For all $x \in \mathcal{R}$, the matrix $I - \hat{M}(x)$ is nonsingular, and its inverse satisfies $(I - \hat{M}(x))^{-1} \geq 0$ componentwise.*

Then the power series:

$$I + \hat{M}(x) + \hat{M}^2(x) + \dots$$

converges componentwise for all nonnegative $x \in \mathcal{R}$, and $[\Psi]_E$ is defined for all $\Psi \in \mathcal{D}^* \Phi'$.

4.1 Minimization of Representations

In this section, we present an algorithm that, given a representation $(C, D, \{M_a : a \in E\})$ for an observable Φ, computes a representation $(C', D', \{M'_a : a \in E\})$ for Φ which is of minimal dimension. The algorithm is based on the following characterization of minimality for representations.

Lemma 8 *Suppose $(C, D(x), \{M_a(x) : a \in E\})$ is an n-dimensional representation of an observable $\Phi \in \mathrm{Obs}(E)$. Then this representation is minimal if and only if neither of the following two (infinite) systems of equations has any nontrivial solutions:*

$$X(\prod_{k=0}^{l-1} M_{a_k}(d_k)) D(d_l) = 0 \qquad C(\prod_{k=0}^{l-1} M_{a_k}(d_k)) Y = 0$$

where l ranges over all nonnegative integers, the a_k range over all elements of E, the d_k range over all nonnegative reals, X is an unknown row vector, and Y is an unknown column vector. Moreover, there exist algorithms for computing a basis for the solution spaces of systems of equations of these forms.

Lemma 9 *Suppose $(C, D(x), \{M_a(x) : a \in E\})$ is an n-dimensional representation of an observable $\Phi \in \mathrm{Obs}(E)$. Suppose the system of all equations of the form:*

$$X(\prod_{k=0}^{l-1} M_{a_k}(d_k)) D(d_k) = 0.$$

has a nontrivial solution X. Let m be the dimension of the solution space. Then Φ has an $(n-m)$-dimensional representation $(C', D'(x), \{M'_a(x) : a \in E\})$, which is effectively computable from the given n-dimensional representation.

An analogous result holds for the system of all equations of the form:

$$C(\prod_{k=0}^{l-1} M_{a_k}(d_k))Y = 0$$

Proof Sketch – Using Lemma 8, we can compute a basis $\mathcal{B} = \{B_1, B_2, \ldots, B_m\}$ (of row vectors) for the solution space S of the above system of equations and a basis $\mathcal{C} = \{C_1, C_2, \ldots, C_{n-m}\}$ for the orthogonal complement $S^\perp$ of S. Assume that the basis $\mathcal{C}$ is orthonormal, which can be ensured using the Gram-Schmidt procedure. Let $P_{S^\perp}$ be the $(n \times (n-m))$-dimensional matrix of the projection of $\mathcal{R}^n$ (row vectors) to $S^\perp$, with respect to the basis $\mathcal{C}$ for $S^\perp$ and the natural basis for $\mathcal{R}^n$. Explicitly, for $1 \leq i \leq n-m$, the ith column of the matrix $P_{S^\perp}$ contains the components of the basis vector C_i. Let $Q_{S^\perp} = P_{S^\perp}^{\mathrm{t}}$, which is the $((n-m) \times n)$-dimensional matrix of the embedding of $S^\perp$ in $\mathcal{R}^n$, with respect to the basis $\mathcal{C}$ for $S^\perp$ and the natural basis for $\mathcal{R}^n$.

We now define $C' = CP_{S^\perp}$, $M'_a(x) = Q_{S^\perp} M_a(x) P_{S^\perp}$, and $D'(x) = Q_{S^\perp} D(x)$. We claim that that $(C', D'(x), \{M'_a(x) : a \in E\})$ is also a representation of Φ. The proof uses the fact that the following relationship holds for all $a \in E$:

$$P_{S^\perp} Q_{S^\perp} M_a(x) P_{S^\perp} = M_a(x) P_{S^\perp}.$$

An analogous, "time-reversed" version of the above argument yields the construction and proof for the second system of equations. ∎

Theorem 5 *There exists an algorithm that, given an n-dimensional representation of an observable $\Phi \in$* Obs(E)*, outputs an m-dimensional representation of Φ, which is minimal.*

Proof Sketch – The algorithm is as follows:

1. Determine the space of solutions X to the system of all equations of the form:

   $$X(\prod_{k=0}^{l-1} M_{a_k}(d_k))D(d_k) = 0.$$

   If this space is nontrivial, use Lemma 9 to produce an n'-dimensional representation of Φ, where $n' < n$.

2. Determine the space of solutions Y to the system of all equations of the form:

   $$C(\prod_{k=0}^{l-1} M_{a_k}(d_k))Y = 0$$

   If this space is nontrivial, use Lemma 9 to produce an n''-dimensional representation of Φ, where $n'' < n'$.

It can be shown that step (2) does not introduce any additional possibility of nontrivial solutions X to the system of the form considered in step (1). Since the n''-dimensional representation resulting from step (2) thus satisfies the conditions of Lemma 8, it is minimal. ∎

5 Example

In this section, we present a very abbreviated example to emphasize the symbolic flavor of the calculations performed using our algorithms.

Let A be the PIOA with $E_A = \{t, a\}$, where t is an internal action and a is an output action, with $Q_A = \{q_0, q_1\}$, where q_0 is the start state, with $\mu_A(q_0, a, q_1) = p$, $\mu_A(q_0, t, q_0) = 1 - p$, and $\mu_A(q_i, a', q_j) = 0$ for all other cases, and with $\delta_A(q_0) = d$ and $\delta_A(q_1) = 0$ (see Figure 2). Let T be the target set consisting of the single string a.

We wish to calculate the expected completion time for A with respect to T. Note that for this simple example, it is possible to determine these quantities by hand, by solving linear equations. In particular, the expected completion time x satisfies the linear equation: $x = 1/d + (1-p)x + p \cdot 0$, which expresses the expected completion time from state q_0 as the sum of the expected holding time in state q_0, plus the sum of the expected delays from the successor states q_0 and q_1 of q_0, weighted respectively by the probability of transitions to these successor states. Solving this equation for x yields the result: $x = 1/dp$. In a similar fashion, the completion probability can be shown to be 1.

We now apply the theory of the preceding sections to provide an alternative calculation of the same quantity. We wish to emphasize that, although the calculations using our methods are more involved in the case of this simple example, the real advantage of our method over the "equation-solving" method will be realized on very large systems having many components. For these cases, the non-compositional equation-solving method will yield an unmanageably large system of equations to be solved (one for each global state), whereas our method can be applied one component at a time, using minimization at each stage, thereby potentially avoiding an explosion in the space required.

Following the theory of the preceding sections, the expected completion time for the above example is given by: $\mathcal{B}_\emptyset^A \Omega_T(0)$. Using Lemma 4, we can obtain a 4-dimensional representation for the observable Ω_T. From this, using Theorem 3, we can com-

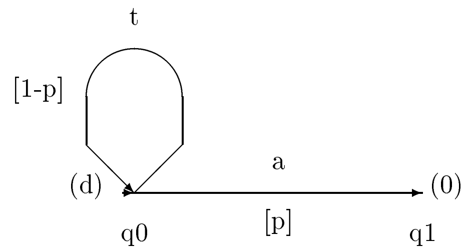

Figure 2: Example PIOA for Calculation of Completion Probability and Expected Completion Time

pute an 8-dimensional representation for $\mathcal{B}^A_{E_A}\Omega_T$. Applying the minimization algorithm of Section 4.1 to this 8-dimensional representation yields the following 3-dimensional representation, which is minimal (the factor of $\sqrt{2}$ appearing below arises from the orthonormalization step which is part of the minimization algorithm):

$$C' = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \qquad D' = \begin{pmatrix} \sqrt{2} \\ 0 \\ 0 \end{pmatrix}$$

$$M'_a(x) = \begin{pmatrix} 0 & 0 & 0 \\ \frac{\sqrt{2}}{2}\frac{dp}{x+d} & 0 & 0 \\ \frac{\sqrt{2}}{2}\frac{dp}{(x+d)^2} & 0 & 0 \end{pmatrix}$$

$$M'_t(x) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \frac{d(1-p)}{x+d} & 0 \\ 0 & \frac{d(1-p)}{(x+d)^2} & \frac{d(1-p)}{x+d} \end{pmatrix}.$$

From this representation of $\mathcal{B}^A_{E_A}\Omega_T$ we now obtain a representation for $\mathcal{B}^A_{\emptyset}\Omega_T$ using Theorem 4. Letting $\hat{M}(x) = M'_a(x) + M'_t(x)$, we can compute:

$$(I - \hat{M}(x))^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ \frac{\sqrt{2}}{2}\frac{dp}{x+dp} & \frac{x+d}{x+dp} & 0 \\ \frac{\sqrt{2}}{2}\frac{dp}{(x+dp)^2} & \frac{d(1-p)}{(x+dp)^2} & \frac{x+d}{x+dp} \end{pmatrix}.$$

The representation for $\mathcal{B}^A_{\emptyset}\Omega_T$ is thus:

$$C'' = C' = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}$$

$$D'' = (I - \hat{M}(x))^{-1}D'(x) = \begin{pmatrix} \sqrt{2} \\ \frac{dp}{x+dp} \\ \frac{dp}{(x+dp)^2} \end{pmatrix}$$

From this, we can calculate the expected completion time.

$$C''D''(0) = \frac{dp}{(0+dp)^2} = \frac{1}{dp}.$$

6 Summary

To summarize, our compositional method for computing expected completion time for a composite system $\|\{A_1, A_2, \ldots, A_n\}$ with respect to a regular target set T is as follows: Use Lemma 4 to obtain a representation for the associated observable Ω_T, use Theorems 3 and 4 to compute a representation for the application of the behavior of A_1 to Ω_T, minimize the result using Theorem 5, then repeat for components A_2, A_3, …, A_n. Apply the resulting representation to the delayed trace (0) to obtain the result. Note that the same method applies to the computation of completion probability with respect to a target set, or in fact any other quantity of interest that can be expressed in terms of the application of the system behavior map to an observable. It is also interesting to note that our method yields a compositional technique for determining reachability of a target set for *non*-probabilistic I/O automata: simply "probabilize" the automaton in question by assigning each transition some nonzero probability, then determine whether the completion probability for the resulting PIOA with respect to the given target set is nonzero.

Note that, under a worst-case analysis, there will exist problem instances for which no minimization is possible, and for these instances our methods will be no better than a non-compositional method. The real test of whether combinatorial explosion is avoided must therefore lie in the implementation of our method (currently in progress) and its application to practical examples. However, there is a strong heuristic reason why we believe minimization is likely to give good results. In our method, we specify at the beginning of the computation (in the form of the particular observable to be used) the particular system property to be analyzed. Each stage of application of the behavior of a component PIOA A therefore amounts to a kind of "partial evaluation," in which the resulting observable expresses only information relevant to the property of interest and the portion of the system analyzed so far. Minimizing the representation obtained at each stage will have the effect of avoiding the accumulation of irrelevant details about the system.

Acknowledgement

The authors wish to acknowledge Xinxin Liu for his valuable participation in a series of technical discussions that contributed substantially to the discovery of Theorem 1.

References

[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*, 1998. To appear.

[BR84] J. Berstel and C. Reutenauer. *Rational Series and Their Languages*, volume 12 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1984.

[CCM97] S. Campos, E. M. Clarke, and M. Minea. Symbolic techniques for formally verifying industrial systems. *Science of Computer Programming*, 29:79–88, 1997.

[CP71] J. W. Carlyle and A. Paz. Realizations by stochastic finite automata. *J. Comput. System Sci.*, 5:26–40, 1971.

[Dij93] Nico M. van Dijk. *Queueing Networks and Product Forms: A Systems Approach*. Series in Systems and Optimization. John Wiley & Sons, 1993.

[GHR93] N. Götz, U. Herzog, and M. Rettelbach. Multiprocessor and distributed system design: the integration of functional specification and performance analysis using stochastic process algebras. In *Proceedings of PERFORMANCE '93*, pages LNCS 729:121–146. Springer-Verlag, 1993.

[Hil96] J. Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.

[How71] Ronald A. Howard. *Dynamic Probabilistic Systems, Volume II: Semi-Markov and Decision Processes*, volume 2 of *Series in Decision and Control*. John Wiley & Sons, 1971.

[LSS94] N. Lynch, I. Saias, and R. Segala. Proving time bounds for randomized distributed algorithms. In *Proceedings of the 13th ACM PODC*, Los Angeles, CA, August 1994.

[MBC84] M. Ajmone Marsan, G. Balbo, and G. Conte. A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems. *ACM Trans. on Computer Systems*, 2:143–172, 1984.

[Pri96] C. Priami. Stochastic π-calculus with general distributions. In M. Ribaudo, editor, *Proceedings of the Fourth Workshop on Process Algebras and Performance Modelling*, Torino, Italy, 1996.

[PS95] A. Pogosyants and R. Segala. Formal verification of time properties for randomized distributed algorithms. In *Proceedings of the 14th ACM PODC*, Ottawa, Ontario, Canada, August 1995.

[Sch61a] M. P. Schützenberger. On a special class of recurrent events. *Annals Math. Stat.*, 32:1201–1213, 1961.

[Sch61b] M. P. Schützenberger. On the definition of a family of automata. *Information and Control*, 4:245–270, 1961.

[SS97] E. W. Stark and S. A. Smolka. Compositional analysis of expected delays in networks of probabilistic I/O automata (full version). Technical report, SUNY at Stony Brook Department of Computer Science, 1997. http://www.cs.sunysb.edu/~stark/REPORTS/delays.ps.Z.

[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.

[Wu96] S.-H. Wu. *Composition and Behaviors of Probabilistic Input/Output Automata*. PhD thesis, SUNY at Stony Brook, 1996.