Probabilistic Timed I/O Automata with Continuous State Spaces*

Sayan Mitra and Nancy Lynch

Computer Science and AI Laboratory,
Massachusetts Inst. of Technology,
32 Vassar Street, Cambridge, MA 02139
{lynch, mitras}@csail.mit.edu

Abstract. We present Piecewise Deterministic Timed I/O Automata (PDTIOA): a new continuous state automaton model that allows both nondeterministic and probabilistic discrete transitions, along with continuous deterministic trajectories. We use a partition of actions, called *tasks* and a *task scheduler* to resolve nondeterministic choice over actions. We define a topology on the set of trajectories and make a key continuity assumption about maximal length of trajectories. Together, these structures enable us to construct a natural probability measure over the space of executions and the space of traces. The resulting PDTIOA framework yields simple notions of external behavior and implementation, and has simple compositionality properties. By introducing local schedulers, we generalize PDTIOAs to allow nondeterministic trajectories and stopping times.

1 Introduction

Probabilistic state machine models provide a mathematical framework for analyzing computing systems that rely on randomization and those that have to contend with uncertainties. Frameworks supporting models with continuous state spaces and continuous probability distributions are necessary to capture interaction of software components with physical processes. For example, a real-time protocol communicating over a channel that delays messages according to a Gaussian distribution or a mobile network routing algorithm that relies on inaccurate position information of participating nodes. Further, in order to abstractly specify concurrent systems, such a modelling framework should allow both nondeterministic and stochastic behavior. For a discussion on the importance of allowing nondeterminism in modelling, we refer the reader to Chapter 4 of [11].

Several continuous state probabilistic models have been proposed in the literature. For instance, Labelled Markov Processes [5] extend traditional transition systems to continuous state spaces and transitions with continuous distributions. In Stochastic Hybrid Systems [12,1] transitions are brought about by discrete or continuous time Markov chains, and the trajectories are described by stochastic

* Supported by the MURI project:DARPA/AFOSR MURI F49620-02-1-0325 grant.

differential equations. In Piecewise Deterministic Markov Processes [6] the time of occurrence and the target states of discrete state transitions are chosen probabilistically; the state evolves deterministically in the intervening time between transitions. The above models do not allow *internal nondeterminism*, that is, choice of an action uniquely determines a transition which gives a probability distribution over the state space. Modeling frameworks that support composition of automata have to resolve *external* nondeterminism: the choice over enabled actions of automata running in parallel. This nondeterminism can be replaced by a race between the automata [8,19], else it can be explicitly resolved by a *scheduler* [3]. Nondeterminism can also be allowed by treating the probabilistic and nondeterministic transitions as separate kinds of objects [11].

Our goal is to develop a modelling framework that allows probabilistic and nondeterministic behavior over continuous state spaces, and to devise a set of proof techniques for analysis of such models. Of particular interest are the techniques, for establishing safety with high probability, almost sure stability [4,14] invariant distributions, and abstraction relationships through probabilistic simulations. Towards this broader goal, in this paper we first develop the basic theory of the *Piecewise Deterministic Timed I/O Automaton (PDTIOA)*. PDTIOA is an interactive state machine model with continuous state space, in which the probabilistic and nondeterministic choices are restricted to the discrete transitions and the trajectories are deterministic. Thus, PDTIOA generalizes the Probabilistic I/O Automaton model (PIOA) of [18,17] by allowing continuous state spaces and distributions. It also generalizes the class Timed I/O Automata (TIOA) [13] with deterministic trajectories, by adding probabilistic transitions. It is worth emphasizing that apart from determinism, we impose only mild continuity restrictions on continuous dynamics of PDTIOAs. In a later section we relax the deterministic trajectories requirement and show how nondeterministic trajectories and stopping times can be included in this modelling framework.

The first step in our development is to define a natural probability measure over the space of executions of a PDTIOA. We use a partition over actions, called *tasks*, and an *oblivious task scheduler* (as in [2]) to resolve the nondeterministic choice over actions. A PDTIOA combined with a task scheduler is an entity that has purely probabilistic behavior, that is, it has no nondeterminism. Thus, we can assign probability measures to its executions. As we are dealing with continuous distributions and state spaces, measurability of sets of executions plays an important role. We cover the essential mathematical groundwork in Section 2 and define a topology on the set of trajectories of a PDTIOA. In Section 3 we introduce the PDTIOA model and construct a σ-algebra on the set of executions and a probability measure on this space. The construction of this measure relies on integrating a certain function over the space of executions. In order for this integral to be well defined, the function must be measurable and this leads us to the key *maximal continuity* assumption about the set of trajectories of a PDTIOA. The assumption roughly states that if τ is the longest trajectory starting from some state x, then the longest trajectories starting from states near x are near τ; here nearness means containment in the open sets of the respective topologies. A probability measure on the space of executions defines a probability measure on the space of traces, provided that the *trace* function is measurable. We show that this is indeed the case; we call a probability distribution over traces a *trace distribution*.

We define parallel composition and hiding operations for PDTIOAs in Section 4, and show that the class of PDTIOA is closed under these operations. We use a simple, but intuitive notion of *external behavior* for PDTIOAs: for a given automaton $\mathcal{A}$, its external behavior is a function that maps each closing environment $\mathcal{E}$ of $\mathcal{A}$ to the set of all possible trace distributions of the composition of $\mathcal{A}$ and $\mathcal{E}$. We show that the *implementation relation* defined in terms of the above notion of external behavior is compositional. Indeed, considering closed automata and using this functional definition of external behavior lets us circumvent some of the difficulties that underlie compositionality in the probabilistic setting. However, viewing external behavior as a mapping from environments as opposed to a set of trace distributions is natural in many applications, including analysis of security protocols [2]. Finally, in Section 5 we obtain the *Generalized PDTIOA* model by relaxing some of the determinism requirements of PDTIOA. This model is capable of capturing nondeterministic trajectories and stopping times using *local schedulers* for automata components. We illustrate the PDTIOA modelling framework with a simple *leaky bucket* example taken from [11]. We give proof sketches for most of the important results; complete proofs appear in Appendix A.

PDTIOAs are similar to the Stochastic Transition Systems (STS) of [3]; the following are important factors distinguishing the two frameworks: (a) general history-dependent, randomized schedulers are used for resolution of nondeterminism in STS[1]. In contrast, in this paper we focus our attention on a simple class of oblivious schedulers for PDTIOAs. (b) In STS there is no notion of time or trajectories, instead state changes through labelled transitions. One obvious approach for modeling a time passage, is to treat it as a transition labeled by the appropriate duration. (c) The above approach, however, leads to a situation where from the trace of any execution one can determine the exact points in time where internal actions occur. In PDTIOA, traces hide information about the time of occurrence of internal actions in an interval, because trajectories separated by an internal action (in an execution) are concatenated in the corresponding trace. Consequently, proving that the *trace* function is measurable turns out to be more involved for PDTIOAs compared to the corresponding proof in the STS setting.

2 Mathematical preliminaries

In this section, we define several operations on trajectories, a topology on a set of trajectories that has desirable properties, and we the review basic concepts from measure theory. We refer the reader to the standard textbooks, e.g. [16,7],

[1] Indeed, all such schedulers are not well behaved and the authors of [3] characterize the subclass that does produce measurable probabilistic executions.

for a comprehensive treatment of the subject; [15] gives a carefully tailored introduction for applications in concurrency theory.

Trajectories. We denote the domain of a function f by $f.dom$. For a set $S \subseteq f.dom$, we write $f \lceil S$ for the restriction of f to S. If f is a function whose range is a set containing Y, then we write $f \downarrow Y$ for the function $g : f.dom \to Y$ defined as $g(c) := f(c) \lceil Y$, for every $c \in g.dom$.

Let $(X, \mathscr{T})$ be a Hausdorff space. Throughout this paper, we will refer to X as the *state space* and elements of X as *states*. We define $\mathbb{T} = \mathbb{R}_{\geq 0} \cup \{\infty\}$ to be the *time axis* and define a topology on $\mathbb{T}$ by declaring the following sets to be open: $(a, b), [0, a), (a, \infty]$ for any $a < b, a, b \in \mathbb{T}$, and any union of segments of this type. For any $J \subseteq \mathbb{T}$ we define $J + t = \{t' + t \mid t' \in J\}$. A *trajectory in* X is a continuous function $\tau : J \to X$, where J is a left closed interval in $\mathbb{T}$ with left endpoint 0. A trajectory with the single point 0 as its domain mapping to the state x, is called the *point trajectory* at x and is written as $\wp(x)$. A trajectory is *finite* and *closed* if $\tau.dom$ is finite and right closed, respectively. The *first state* of τ, $\tau.fstate$ is $\tau(0)$. The *limit time* of a trajectory τ is $\sup\{\tau.dom\}$ and is written as $\tau.ltime$. If τ is closed and finite then the *limit state* of τ, $\tau.lstate$, is $\tau(\tau.ltime)$.

Given a trajectory τ and $t \in \mathbb{T}$, the function $(\tau + t) : (\tau.dom + t) \to X$ is defined as $(\tau + t)(t') := \tau(t' - t)$, for each $t' \in (\tau.dom + t)$. Given two trajectories τ_1 and τ_2, τ_1 is a *prefix* of τ_2, written as $\tau_1 \leq \tau_2$, if $\tau_1 = \tau_2 \lceil \tau_1.dom$. Also, τ_1 is a *suffix* of τ_2 if $\tau_1 = (\tau_2 \lceil [t, \infty]) - t$, for some $t \in \tau_2.dom$. If τ_1 is a closed trajectory with $\tau_1.ltime = t$ and $\tau_2.fstate = \tau_1.lstate$, then the function $\tau_1 \frown \tau_2 : \tau_1.dom \cup (\tau_2.dom + t) \to X$ is defined as $\tau_1(t)$ if $t \leq u$ and $\tau_2(t - u)$ otherwise. Given a set of trajectories $\mathcal{T}$, for any $x \in X$ we define $\mathcal{T}(x) = \{\tau \in \mathcal{T} \mid \tau.fstate = x\}$. A set of trajectories $\mathcal{T}$ is said to be *deterministic* if for any $x \in X$, if $\tau_1, \tau_2 \in \mathcal{T}(x)$ then either $\tau_1 \leq \tau_2$ or $\tau_2 \leq \tau_1$. Given a set of trajectories $\mathcal{T}$ for X, we define a function $max_{\mathcal{T}} : X \to 2^{\mathcal{T}}$ as $max_{\mathcal{T}}(x) := \{\tau \in \mathcal{T}(x) \mid \nexists \tau' \in \mathcal{T}(x), \tau < \tau'\}$. The next proposition follows immediately from observing that if $\tau_1, \tau_2 \in max(x)$ and $\tau_1 \neq \tau_2$, then either $\tau_1 < \tau_2$ or $\tau_2 < \tau_1$.

Proposition 1. *If $\mathcal{T}$ is deterministic, then $max(x)$ is singleton for any $x \in X$.*

In this paper, all sets of trajectories are closed under prefix, suffix, concatenation and except for Section 5, they are also deterministic. For notational convenience we redefine $max : X \to \mathcal{T}$ as the function that gives the single maximal trajectory from state x. Given $\mathcal{T}$, we define the set $\bar{X}_{\mathcal{T}}$, such that $x \in \bar{X}_{\mathcal{T}}$ if and only if $max(x)$ is finite and closed. We define the functions $maxtime_{\mathcal{T}} : \bar{X}_{\mathcal{T}} \to \mathbb{R}_{\geq 0}$ and $maxstate_{\mathcal{T}} : \bar{X}_{\mathcal{T}} \to X$ as $maxtime_{\mathcal{T}}(x) := max_{\mathcal{T}}(x).ltime$ and $maxstate_{\mathcal{T}}(x) := max(x).lstate$, respectively. If the set $\mathcal{T}$ is clear from context, we drop the suffix $\mathcal{T}$ from max, $maxstate$ and $maxtime$.

A topology on the set of trajectories. In developing the theory we will need to impose certain continuity assumptions on the functions defined earlier on the set of trajectories. Therefore, we have to define a topology on this set. A natural topology on a given set of trajectories $\mathcal{T}$ is generated by the basis elements of the form $B_\tau := \{\tau' \mid \tau \leq \tau'\}$, for each $\tau \in \mathcal{T}$. The topology generated by this basis, however, does not have the desirable Hausdorff property.

We define the a basis element to be a set of trajectories contained within a finite concatenation of "open cylinders" of rational lengths. Intuitively, a basis element containing a trajectory τ, over-approximates the states reached by τ. A finer over-approximation is be obtained by a smaller basis element, one defined by shorter and smaller open cylinders.

Definition 1. *Let $(X, \mathscr{T})$ be a Hausdorff space and $\mathcal{T}$ be a set of trajectories for X. A* trajectory base *is a finite sequence $\theta = (q_1, U_1)(q_2, U_2) \dots (q_m, U_m)$, where each $q_i \in \mathbb{Q}_{\geq 0} \cup \{\infty\}$, $q_i < q_{i+1}$ and U_i is an open set in $\mathscr{T}$. A* trajectory basis element *B_θ corresponding to the above trajectory base θ is the set of trajectories:*

$$B_\theta = \begin{cases} \{\tau \in \mathcal{T} \mid \tau.dom \subset [0, q_1],\ \forall t \in \tau.dom, \tau(t) \in U_1\}, & \text{if } m = 1, \\ \left\{ \tau \in \mathcal{T} \middle| \begin{array}{l} q_{m-1} < \tau.ltime < q_m,\ \forall\ t \in [0, q_1], \tau(t) \in U_1, \\ \forall i \in \{1, \dots, m-1\}, \forall\ t \in \tau.dom \cap (q_i, q_{i+1}], \tau(t) \in U_{i+1} \end{array} \right\} & \text{if } m > 1. \end{cases}$$

Any trajectory $\tau \in \mathcal{T}$ is in the basis element $B_{(\infty, X)}$. For any $\tau \in B_{\theta_1} \cap B_{\theta_2}$, we can construct a basis element θ_3 by sorting the rationals in θ_1 and θ_2 and intersecting the appropriate open sets, such that $\tau \in B_{\theta_3} \subseteq B_{\theta_1} \cap B_{\theta_2}$. Thus the collection of trajectory bases in Definition 1 defines a basis for a topology on $\mathcal{T}$. In fact, the generated topological space $(\mathcal{T}, \mathscr{T})$ is a Hausdorff space. A proof of this fact is given in Appendix A.1. Having defined a topological space of trajectories, we then state our key continuity assumption.

Theorem 1. *The collection $\mathscr{B}$ of all trajectory basis elements is a basis for a topology on $\mathcal{T}$. If the topology generated is $\mathscr{T}$ then $(\mathcal{T}, \mathscr{T})$ is a Hausdorff space.*

Definition 2. *A set of trajectories $\mathcal{T}$ is said to be* maximally continuous *if the functions max and maxstate are continuous.*

Continuity of *maxtime* follows from continuity of *max*. Intuitively, maximal continuity states that the maximal trajectories starting from nearby states are close, and also that if the maximal trajectories starting from nearby states are closed and finite then the limiting states of those trajectories are close.

Measurability and measures. We denote a measurable space by $(X, \mathcal{F}_X)$, where X is a set and $\mathcal{F}_X$ is a σ-algebra over X. Given a topological space $(X, \mathscr{T})$, there exists a smallest σ-algebra containing $\mathscr{T}$ and it is called the *Borel σ-algebra*. The *product* of two measurable spaces $(X, \mathcal{F}_X)$ and $(Y, \mathcal{F}_Y)$ is defined as the measurable space $(X \times Y, \mathcal{F}_X \otimes \mathcal{F}_Y)$, where $\mathcal{F}_X \otimes \mathcal{F}_Y$ is the smallest σ-algebra generated by sets of the form $A \times B = \{(x, y) \mid x \in A, y \in B\}$, for all $A \in \mathcal{F}_X, B \in \mathcal{F}_Y$.

A *measure* over $(X, \mathcal{F}_X)$ is a function $\mu : \mathcal{F}_X \to \mathbb{R}_{\geq 0}$, such that $\mu(\emptyset) = 0$ and for every countable collection of disjoint sets $\{A_i\}_{i \in I}$ in $\mathcal{F}_X$, $\mu(\cup_{i \in I} A_i) = \sum_{i \in I} \mu(A_i)$. A *probability measure* (resp. *sub-probability measure*) over $(X, \mathcal{F}_X)$ is a measure μ such that $\mu(X) = 1$ ($\mu(X) \leq 1$). The set of probability measures and the set of sub-probability measures over $(X, \mathcal{F}_X)$ are denoted by $\mathcal{P}(X, \mathcal{F}_X)$ and $\mathcal{SP}(X, \mathcal{F}_X)$. A function $f : (X, \mathcal{F}_X) \to (Y, \mathcal{F}_Y)$ is said to be *measurable* if $f^{-1}(E) \in \mathcal{F}_X$ for every $E \in \mathcal{F}_Y$. The *indicator function* of a set $A \subseteq X$, is defined as $I_A(x) = 1$ if $x \in A$, and 0 otherwise. If A is a measurable set, then I_A is a measurable function. If $f : (X, \mathcal{F}_X) \to (Y, \mathcal{F}_Y)$ is a measurable function, and μ is a measure on X, then the *image measure of* μ *under* f is a measure φ on Y defined as $\varphi(E) = \mu(f^{-1}(E))$, for each $E \in \mathcal{F}_Y$.

A collection $\mathscr{C}$ of subsets of X, is a *semi-ring* if $X, \emptyset \in \mathscr{C}$, $A, B \in \mathscr{C}$ implies that $A \cap B \in \mathscr{C}$, and $A, B \in \mathscr{C}$ implies that there exists a finite collection of disjoint sets $\{C\}_{i=1}^n$ in $\mathscr{C}$ such that $A \setminus B = \cup_{i=1}^n C_i$. We will use the following theorem to constructing measures of the space of executions of an automaton:

Theorem 2. *A probability measure defined over a semi-ring* $\mathscr{C}$ *can be uniquely extended to a probability measure over the* σ*-algebra generated by* $\mathscr{C}$*.*

In constructing measures over the space of executions of a PDTIOA, we have to integrate over the space of probability distributions over the state space X, therefore we need to define a σ-algebra over $\mathcal{P}(X, \mathcal{F}_X)$. For this, we use the following construction due to Giry [9]: for each $A \in \mathcal{F}_X$, let the function $p_A : \mathcal{P}(X, \mathcal{F}_X) \to [0, 1]$ be defined as $p_A(\mu) = \mu(A)$. The σ-algebra on $\mathcal{P}(X, \mathcal{F}_X)$, then is the smallest σ-algebra such that all p_A's are measurable.

3 Piecewise Deterministic Timed I/O Automata

In this section we present the basic theory of *Piecewise Deterministic Timed I/O Automaton (PDTIOA)*.

Definition 3. *A* Piecewise Deterministic TIOA(PDTIOA) *is a tuple* $\mathcal{A} = ((X, \mathcal{F}_X), \bar{x}, A, \mathcal{R}, \mathcal{D}, \mathcal{T})$ *where: (1)* $(X, \mathcal{F}_X)$ *is a measurable space;* X *is Hausdorff. (2)* $\bar{x} \in X$ *is the* start state*. (3)* A *is a countable set, called the set of* actions*.* A *is partitioned into* internal H*,* input I *and* output O *actions. The set* $L = O \cup H$ *is called the set of* local actions *and the set* $E = O \cup I$ *is called the set of* external actions*. (4)* $\mathcal{R}$ *is an equivalence relation on* L*; equivalence classes of* $\mathcal{R}$ *are called* tasks*. (5)* $\mathcal{D} \subseteq X \times A \times \mathcal{P}(X, \mathcal{F}_X)$ *is called the set of* transitions *and it satisfies:* **M1** *For all* $a \in A$*,* $\{x \mid \exists \mu, (x, a, \mu) \in \mathcal{D}\} \in \mathcal{F}_X$*. (6)* $\mathcal{T}$ *is a set of maximally continuous, deterministic trajectories for* X *that is closed under prefix, suffix, concatenation and contains* $\wp(x)$ *for every* $x \in X$*.* **M2** *The set* $\mathcal{T}$ *should be such that* $\bar{X}_{\mathcal{T}}$ *is measurable.*

In addition, $\mathcal{A}$ *satisfies the following axioms:*

D0 (Input action enabling) *For all* $x \in X$*,* $a \in I$*, there exists* $\mu \in \mathcal{P}(X, \mathcal{F}_X)$ *such that* $(x, a, \mu) \in \mathcal{D}$*.*

D1 (Time-action determinism) *For all* $x \in X$ *at most one of the following conditions may hold: (1) there exists* $a \in L$ *that is enabled at* x *(2) there exists a non-point trajectory* $\tau \in \mathcal{T}$ *with* $\tau.fstate = x$*.*

D2 (Task determinism) *For all $x \in X$, if $a_1, a_2 \in T$, for some task T and $\mu_1, \mu_2 \in \mathcal{P}(X, \mathcal{F}_X)$, such that $(x, a_1, \mu_1) \in \mathcal{D}$ and $(x, a_2, \mu_2) \in \mathcal{D}$ then $a_1 = a_2$ and $\mu_1 = \mu_2$.*

D3 *Over any finite time interval at most finite internal actions may occur.*

An action a is said to be *enabled* at x if there exists $\mu \in \mathcal{P}(X, \mathcal{F}_X)$ such that $(x, a, \mu) \in \mathcal{D}$. In short, we write this as $x \xrightarrow{a} \mu$. If $x \xrightarrow{a} \mu$, then we write $\mu_{a,x}$ in place of μ and if a single action is enabled at x then we write μ_x instead of $\mu_{a,x}$. We define the function *enabled* : $X \rightarrow 2^L$ as *enabled*$(x) := \{a \in L \mid a$ is enabled at $x\}$, for any $x \in X$. It can be checked easily that *enabled* is a measurable function.

Discussion on PDTIOA. **D1-2** make $\mathcal{A}$ piecewise deterministic. That is, at any state x either some local action can occur or some non-zero amount of time can elapse, not both. In the former case, for any given task a unique transition is enabled. and in the latter case, a trajectory is uniquely determined by the amount of time that elapses. **D1** is similar to the maximal progress assumption found in real-time process algebras, e.g. [10]. It prevents, for example, an action to remain enabled while time elapses. In Section 5 we relax this assumption to allow both nondeterministic trajectories and stopping times. **D0** is a non-blocking assumption standard in I/O automata literature. **D3** helps us avoid certain technical difficulties is constructing the set of traces from a given execution. The assumption that the set of actions A is countable is a non-essential simplification with little loss of generality. In PDTIOAs time passage is modeled by trajectories, so this assumption conforms with the conventional use of labelled transitions. Wherever necessary, we assume that A is equipped with the discrete topology and that every subset of A is measurable. The closure assumptions on $\mathcal{T}$ hold trivially for time invariant systems that we are interested in. The determinism assumption is later relaxed to include nondeterministic trajectories but we still exclude trajectories where stochastic choices are made continuously over a an interval of time, e.g., Brownian motion. This will be investigated in the future.

3.1 Executions and traces

An *execution fragment* of an PDTIOA $\mathcal{A}$ is an alternating sequence of actions and trajectories $\alpha = \tau_0 a_1 \tau_1 a_2 \ldots$, where each $\tau_i \in \mathcal{T}$, each $a_i \in A$ and a_i is enabled at $\tau_{i-1}.lstate$. The *first state* of an execution fragment α, $\alpha.fstate$, is $\tau_0.fstate$. An execution fragment α is an *execution* of $\mathcal{A}$ if $\alpha.fstate = \bar{x}$. The *length* of a finite execution fragment α is the number of actions in α. An execution fragment is *closed* if it is a finite sequence and last trajectory is closed. Given a closed execution fragment $\alpha = \tau_0 a_1 \ldots \tau_n$, its *limit state*, $\alpha.lstate$, is $\tau_n.lstate$ and its *limit time* is defined as $\sum_i^n \tau_i.ltime$. The next proposition is a direct consequence of **D1**.

Proposition 2. *Non-final trajectories in an execution are finite, closed and maximal.*

The *trace* of an execution α is represents its externally visible part, namely the external actions and time passage; it is inductively defined as:

$$\begin{aligned} trace(\alpha) &= \tau.ltime \quad \text{if } \alpha = \tau, \\ trace(\alpha a \tau) &= \begin{cases} trace(\alpha)a\ \tau.ltime & \text{if } a \in E, \\ trace(\alpha')\ (\tau' \frown \tau).ltime & \text{where } \alpha = \alpha'\tau', \text{ otherwise.} \end{cases} \end{aligned}$$

Informally, the trace of α is obtained by removing all internal actions from α, concatenating the resulting consecutive trajectories in α, and replacing all the trajectories with their limit times. Concatenating consecutive trajectories hides information about the point of occurrence of internal actions in the trace of a PDTIOA. The importance of this will be clear in Section 3.4 where we prove the measurability of the *trace* function. Note that this definition for trace differs significantly from the one used in [3].

We denote the set of execution fragments, the set of executions, and the set of traces of PDTIOA $\mathcal{A}$ by $\mathsf{Frags}_{\mathcal{A}}$, $\mathsf{Execs}_{\mathcal{A}}$ and $\mathsf{Traces}_{\mathcal{A}}$. The set of finite fragments, finite executions and finite traces are denoted by $\mathsf{Frags}^*_{\mathcal{A}}$, $\mathsf{Execs}^*_{\mathcal{A}}$ and $\mathsf{Traces}^*_{\mathcal{A}}$.

Example 1. The *leaky bucket* access control mechanism is used in ATM networks to negotiate the data-rate between a sender and the service provider. As data arrives at the ATM switch they flow into a 'bucket' which drains at bucket rate. If the data arrives faster than the bucket is draining eventually the bucket will overflow. The leaky bucket model of Figure 1 is the PDTIOA version of the mechanism presented in [11].

Automaton ATM has two queues *dataQ* and *tokenQ*, a clock $clock_A$ and a deadline variable $delay_A$. The *dataQ* buffers bits arriving with the Data action. Data bits have to wait in the buffer until service is available. Availability of service is represented by *tokens* which is obtained through the input action Token. If *dataQ* and *tokenQ* are both nonempty, then one bit of data is ready to be sent. The service time for sending is exponentially distributed with parameter λ_A. This is captured by the $Reset_A$ subroutine, which resets $clock_A$ and chooses the next sending delay from an exponential distribution with parameter λ_A. Notice that if *dataQ* (or *tokenQ*) empties out then $delay_A$ is set to ∞ until new data (resp token) arrives.

The arrival of data bits and tokens through Data and Token actions are modeled by two separate automata $\mathsf{DATASRC}$ and $\mathsf{TOKENSRC}$ (see Figure 2 in Appendix B). Following the model in [11], we assume that data bits and tokens possess exponentially distributed inter-arrival times, given by rate parameters λ_D and λ_T and that all the clocks are reset after each discrete transition. A typical execution of ATM is a sequence $\alpha = \tau_0\ \mathsf{Data}(1)\ \tau_1\ \mathsf{Token}\ \tau_2\ \mathsf{Token}\ \tau_3\ \mathsf{Send}(1)\ \tau_4\ \mathsf{Data}(0)\ \tau_5 \ldots$, were each τ_i, $i \in \{0, \ldots, 5\}$, is a trajectory over which $clock_A$ increase monotonically at a constant rate of 1 and all other variables remain constant.

3.2 σ-Algebra of Executions and Traces

In order to construct a probability measure over the set of executions of a given PDTIOAs $\mathcal{A}$, we have to first define the measurable sets in Execs_A. In

```
ATM($\lambda_A$) where $\lambda_A \in \mathbb{R}_+$
Variables:
  $clock_A : \mathbb{R}_+$ initially $0$
  $delay_A : \mathbb{R}_+$ initially $\infty$
  $dataQ$, $tokenQ$: queue of $\{0,1\}$ initially $\perp$

Actions:
  output Send($m$) $m \in \{0,1\}$
  input Token, Data($m$) $m \in \{0,1\}$

Transitions:
  Send($m$)
  pre  $clock_A = delay_A \wedge m = head(dataQ)$
  eff dequeue($dataQ$); dequeue($tokenQ$)
    if $\neg$ $empty(dataQ) \wedge \neg$ $empty(tokenQ)$
      $Reset_A$
    else $delay_A := \infty$; $clock_A := 0$ fi

  Data($m$)
  eff $dataQ :=$ append($dataQ$, $m$)
    if $\neg$ $empty(tokenQ)$ $Reset_A$ fi

  Token
  eff $tokenQ :=$ append($tokenQ$, $1$)
    if $\neg$ $empty(dataQ)$ $Reset_A$ fi

Trajectories:
  Trajdef normal
  invariant $clock_A \leq delay_A$
  evolve $d(clock_A) = 1$

Tasks: {Send}

Subroutine $Reset_A$:
  $delay_A :=$ choose $Exp(\lambda_A)$;
  $clock_A := 0$
```

Fig. 1. Automaton ATM. The $Reset_A$ resets $clock_A$ and sets $delay_A$ according to exponential distribution with parameter λ_A.

the case of probabilistic automata with discrete state spaces [18,17,2], the standard approach is to define the σ-algebra as the collection of sets of the form $E_\alpha := \{\alpha' \mid \alpha \text{ is a prefix of } \alpha'\}$. Then, one can define the probability of the set E_α as the probability of performing α, which can computed from the probabilistic transitions in α. This approach does not work directly when the transitions give a continuous probability distribution over the state space because the probability of occurrence of any particular finite sequence of transitions is typically 0. We follow the approach used in [3]; instead of considering a set of executions that extend a *single* prefix, we consider a set containing executions that extend an any prefix from an *uncountable set* of prefixes.

Definition 4. *A* base *is a finite sequence of the form $\Lambda = X_0 R_0 X_1\ A_1\ X_2 R_1 X_3\ A_2 \ldots X_{2m-1} A_m X_{2m}$, where for every $i \in \{0, \ldots, 2m\}$, $X_i \in \mathcal{F}_X$, for every $i \in \{1, \ldots, m\}$, $A_i \subseteq A$ and for every $i \in \{0, \ldots, m-1\}$, R_i is a measurable set in $\mathbb{R}_{\geq 0}$. The* length *of a base is the number of sets of actions in the sequence. The* basic set *corresponding to a base Λ is a set of execution fragments of $\mathcal{A}$ defined as follows:*

$$C_\Lambda = \{\ \tau_0 a_1 \tau_1 \ldots a_m \alpha \in \mathsf{Frags}_{\mathcal{A}} \mid \forall i \in \{0, \ldots, m-1\}\quad \tau_i.fstate \in X_{2i},\\ \tau_i.ltime \in R_i, \tau_i.lstate \in X_{2i+1},\quad \forall i \in \{1, \ldots, m\},\quad a_i \in A_i\}. \tag{1}$$

Lemma 1. *The collection $\mathscr{C}$ of all basic sets of $\mathcal{A}$ is a semi-ring.*

Appendix A.1 contains a proof of the above lemma. We denote the σ-algebra generated by $\mathscr{C}$ as $\mathcal{F}_{\mathsf{Execs}\mathcal{A}}$ and the measurable space of executions of $\mathcal{A}$ by $(\mathsf{Execs}_{\mathcal{A}}, \mathcal{F}_{\mathsf{Execs}_{\mathcal{A}}})$. Theorem 2 states that a measure defined on a semi-ring uniquely extends to a measure on the σ-algebra generated by the semi-ring. The above result will allow us to construct measure on $\mathcal{F}_{\mathsf{Execs}}$ by defining their value for the basic sets. Analogous to Definition 4, we define basic sets for traces and show that the collection of these basic sets is a semi-ring in $\mathsf{Traces}_{\mathcal{A}}$.

Definition 5. *A* trace base *is a finite sequence of the form $\Lambda = R_0E_1 \ldots E_n$ where $\forall i \in \{0, \ldots, n-1\}$, R_i is a measurable set in $\mathbb{R}_{\geq 0}$ and and $\forall j \in \{1, \ldots, n\}$, $E_j \subseteq E$. The* length *of a trace base is the number of sets of actions in the sequence. The* trace basic set *corresponding to the base Λ is a set of traces of $\mathcal{A}$ defined as: $C_\Lambda = \{r_0a_1r_1 \ldots a_n\beta \in \mathsf{Traces}_\mathcal{A} \mid \forall i \in \{0, \ldots, n\} \quad r_i \in R_i, a_i \in E_i\}$.*

Lemma 2. *The collection $\mathscr{D}$ of all trace basic sets of $\mathcal{A}$ is a semi-ring.*

The σ-algebra $\mathcal{F}_{\mathsf{Traces}}$ on the set of traces of $\mathcal{A}$ is defined as the σ-algebra generated by the collection of trace basic sets; we denote the measurable space of traces by $(\mathsf{Traces}_\mathcal{A}, \mathcal{F}_{\mathsf{Traces}_\mathcal{A}})$.

3.3 A probability measure over executions

A PDTIOA $\mathcal{A}$ is *closed* if it has no input actions. A PDTIOA $\mathcal{A}$ is a nondeterministic state machine, that is, it is possible for multiple actions to be enabled at a given state. In order to obtain purely probabilistic executions of closed $\mathcal{A}$ we have to resolve nondeterminism. Our approach is to use the task mechanism and an oblivious scheduler as in [2]. A *task schedule* for a closed PDTIOA $\mathcal{A} = ((X, \mathcal{F}_X), \bar{x}, A, \mathcal{R}, \mathcal{D}, \mathcal{T})$ is simply a finite or infinite sequence $\rho = T_1T_2 \ldots$ of tasks in $\mathcal{R}$. Given a task schedule $\rho = T_1T_2 \ldots$ for closed PDTIOA $\mathcal{A}$, we construct a measure φ_ρ on $(\mathsf{Execs}_\mathcal{A}, \mathcal{F}_{\mathsf{Execs}_\mathcal{A}})$ by inductively defining its values for the basic sets. The induction is on the length of the base Λ.

$$\varphi_\rho(C_X) = \begin{cases} 1 & if\ \bar{x} \in X \\ 0 & otherwise \end{cases}$$

$$\varphi_\rho(C_{\Lambda RX}) = \int_{\alpha \in \Lambda \cap s \in \bar{X}} I_R(maxtime(s)) I_X(maxstate(s)) \varphi_\rho(d\alpha), \tag{2}$$

$$\varphi_{T\rho'}(C_{\Lambda AX}) = \begin{cases} \int_{\alpha \in \Lambda} I_{A \cap T}(enabled(s)) \mu_s(X) \varphi_{\rho'}(d\alpha), & \text{if } \rho = T\rho' \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Here $s = \alpha.lstate$.

Note that the integral in Equation (2) is restricted to $\alpha.lstate \in \bar{X}_\mathcal{T}$ and $maxstate$ is well defined and continuous in this set. Since R and X are measurable sets, I_R and I_X are measurable functions. As $maxstate$ and $maxtime$ are continuous, the compositions $I_R \circ maxtime$ and $I_X \circ maxstate$ are both measurable and therefore the integral in Equation (2) is well defined. Likewise, the integral in Equation (3) is well defined because $A \cap T$ is a measurable set and $enabled \circ I_{A \cap T}$ is a measurable function; and measurability of $\mu_s(X)$ follows from **D2** and the Giry construction described in Section 2. Note that, if length of Λ is l then $\varphi_\rho(C_\Lambda) > 0$ only if length of $\rho \geq l$ and the probability $\varphi(C_\Lambda)$ depends on only on the first l tasks in ρ.

Proposition 3. *Given a PDTIOA $\mathcal{A}$ and a task schedule ρ for $\mathcal{A}$, φ_ρ is a probability measure on $(\mathsf{Execs}_\mathcal{A}, \mathcal{F}_{\mathsf{Execs}_\mathcal{A}})$.*

Given any task schedule ρ the corresponding measure φ_ρ is a probability distribution over $\mathsf{Exec}_\mathcal{A}$, and is called a *probabilistic execution* of $\mathcal{A}$.

3.4 Probability measure on Traces

We will prove that the function $trace : \mathsf{Execs} \to \mathsf{Traces}$ is measurable. This is a necessary property because it allows us to define a measure on the the space of traces $(\mathsf{Traces}, \mathcal{F}_{\mathsf{Traces}})$ corresponding to any measure φ on the space of executions $(\mathsf{Execs}, \mathcal{F}_{\mathsf{Execs}})$. We need two ingredients for this proof: Lemma 3 establishes that for any function $f : \mathsf{Execs} \to \mathsf{Traces}$, to be measurable it is sufficient to show that $f^{-1}(C_\Gamma) \in \mathcal{F}_{Execs}$, for every trace base Γ chosen from a restricted class of bases; Proposition 4 gives us a procedure to partition an interval in $\mathbb{R}_{\geq 0}$, arbitrarily finely, with rational endpoints. This enables us to reconstruct executions from a given trace by accurately inserting internal actions over an interval of time.

Lemma 3. *Consider a function $f : (\mathsf{Execs}, \mathcal{F}_{\mathsf{Execs}}) \to (\mathsf{Traces}, \mathcal{F}_{\mathsf{Traces}})$. If $f^{-1}(C_\Gamma) \in \mathcal{F}_{\mathsf{Execs}}$ for every trace base Γ of the form $[0, b_0)E_1[0, b_1)E_2 \dots E_n$, where $b_i \in \mathbb{R}_{\geq 0}$ and $E_i \subseteq E$ for each i, then f is measurable.*

Proof. We define $\mathscr{C} = \{C \subseteq \mathsf{Traces} \mid f^{-1}(C) \in \mathcal{F}_{\mathsf{Execs}}\}$. First we show that $\mathscr{C}$ is a σ-algebra on Traces.

1. $f^{-1}(\mathsf{Traces}) = \mathsf{Execs} \in \mathcal{F}_{\mathsf{Execs}}$, therefore $\mathsf{Traces} \in \mathscr{C}$.
2. For any $C \in \mathscr{C}$, $f^{-1}(\mathsf{Traces} \setminus C) = \mathsf{Execs} \setminus f^{-1}(C) \in \mathcal{F}_{\mathsf{Execs}}$.
3. For any $C_1, C_2 \in \mathscr{C}$, $f^{-1}(C_1 \cup C_2) = f^{-1}(C_1) \cup f^{-1}(C_2) \in \mathcal{F}_{\mathsf{Execs}}$.

Now, consider any trace base $\bar{\Gamma}_1 = [0, b)E_1$ of unit length, where $E_1 \subset E$ and $b \in \mathbb{R}_{\geq 0}$. Assume that $f^{-1}(C_{\bar{\Gamma}_1}) \in \mathcal{F}_{\mathsf{Execs}}$ for every $\bar{\Gamma}_1$ of the above special form. Choose a sequence of real numbers $\{b_n\}_{n=1}^{\infty}$ such that $b_{n+1} < b_n$ and $b_n \to b$ as $n \to \infty$. From the hypothesis we know that for each n, $C_{[0,b_n)E_1} \in \mathscr{C}$. As

$$C_{(b,\infty]E_1} = \bigcup_{n=1}^{\infty} C_{[b_n,\infty]E_1} = \bigcup_{n=1}^{\infty} C_{[0,b_n)^c E_1}$$

and $\mathscr{C}$ is a σ-algebra it follows that $C_{(b,\infty]E_1} \in \mathscr{C}$. The same holds true for basic sets of the form $C_{(a,b)E_1} = C_{[0,b)E_1} \cap C_{(a,\infty]E_1}$. Since every measurable set in $\mathbb{R}_{\geq 0}$ is a countable union of segments of the types $[0, b), (b, \infty]$, and (a, b), we have proved that for any trace base Γ_1 of unit length, $C_{\Gamma_1} \in \mathscr{C}$ which implies that $f^{-1}(C_{\Gamma_1}) \in \mathcal{F}_{\mathsf{Execs}}$.

Following the same steps of reasoning as above, we show that if $f^{-1}(C_{\bar{\Gamma}})$ is in $\mathcal{F}_{\mathsf{Execs}}$ for every $\bar{\Gamma}$ of the form $[0, b_0)E_1[0, b_1)E_2 \dots E_n$, then in fact for every trace base Γ, $f^{-1}(C_\Gamma)$ is in $\mathcal{F}_{\mathsf{Execs}}$. Since every set in $\mathcal{F}_{\mathsf{Traces}}$ can be expressed as a countable union of the basic sets, the result follows immediately.

Here we informally define the function $rcuts_k$, which takes an interval J in $\mathbb{R}_{\geq 0}$ and is well defined for all $k \in \mathbb{N}$ (see Definition 12 in Appendix A.2). The $rcuts_1$ function partitions a given interval (a, b) into two pieces at a rational point q, such that the point q lies somewhere in the middle third of (a, b). Likewise, $rcuts_k$ partitions (a, b) into 2^k pieces with rational end points. A k-partition of (a, b) is obtained by selecting pieces from any $rcut_j(a, b)$ such that $j \leq k$. Using the $rcuts_k$ function we get the following proposition:

Proposition 4. *Given an interval I and a real number $r \in I$, there exists a k-partition of I with an endpoint that coincides with r, as $k \to \infty$.*

Lemma 4. *For every trace base of the form $\Gamma = [0, b_0)E_1[0, b_1)E_2 \dots E_n$, where $b_i \in \mathbb{R}_{\geq 0}$, $E_i \subseteq E$ for each i, $trace^{-1}(C_\Gamma) \in \mathcal{F}_{\mathsf{Execs}}$.*

Proof. Let us consider a trace base $\Gamma_1 = [0, b_0)E_1$ of unit length. Let $P_k = ([0, q_{k1}], [q_{k1}, q_{k2}], \dots, [q_{km_k}, b_0))$ be a particular k-partition of $[0, b)$. We define a set of base corresponding to P_k as:

$$\Lambda(P_k) = \bigcup_{j_1} \bigcup_{j_2} \dots \bigcup_{j_{m_k}} X[0, q_{k1}](XHX\{0\})^{j_1} X[q_{k1}, q_{k2}](XHX\{0\})^{j_2} X \dots$$
$$\dots (XHX\{0\})^{j_{m_k}} X[q_{km_k}, b)E_1$$

Each sequence on the right hand side is a valid execution base[2] and by definition the corresponding basic sets of executions are in $\mathcal{F}_{\mathsf{Execs}}$. The set of k-partitions $S_k([0, b_0))$ is a finite set, so the set of executions

$$\mathscr{C} = \bigcup_{k=1}^{\infty} \bigcup_{P_k \in S_k([0,b_0))} \bigcup_{B \in \Lambda(P_k)} C_B \tag{4}$$

is a countable union of sets in $\mathcal{F}_{\mathsf{Execs}}$ and therefore is itself in $\mathcal{F}_{\mathsf{Execs}}$. In order to show that $trace^{-1}(C_{\Gamma_1}) \in \mathcal{F}_{\mathsf{Execs}}$ it suffices to show that $trace^{-1}(C_{\Gamma_1}) = \mathscr{C}$. It is easy to see that if an execution $\alpha \in \mathscr{C}$ then $trace(\alpha) \in C_{\Gamma_1}$. For the reverse direction, consider an execution with a single internal action $\alpha = \tau_0 h_1 \tau_1 e \beta$, where for $i \in \{0, 1\}$, $\tau_0.ltime = r \leq \tau_1.ltime < b_0$, for some $r \in \mathbb{R}_{\geq 0}$, $h_1 \in H$, $e \in E$, and β is an execution fragment of $\mathcal{A}$. Clearly $trace(\alpha)$ is in C_{Γ_1}; now we show that α is also in $\mathscr{C}$. From Proposition 4 we know that as there exists a sequence of intervals $[0, r_k] \in S_k([0, b_0))$ such that $r_k \to r$ as $k \to \infty$. Thus, there exists a base $\Lambda = X[0, r]XHX[r, b_0)$ in $\bigcup_{k=1}^{\infty} \Lambda(P_k)$ such that $\alpha \in C_\Lambda \subseteq \mathscr{C}$.

For an execution α with any finite number of internal actions preceding the external action e, we show that $\alpha \in \mathscr{C}$ by induction on the number of internal actions[3].

For an arbitrary length trace base $\Gamma = [0, b_0)E_1[0, b_1)E_2 \dots E_n$, the same reasoning works. The partitions of the $[0, b_i)$ intervals are independent and therefore we get a countable union of n countable sets, where each set has the same form as $\mathscr{C}$ above.

From Lemmas 3 and 4 we obtain the following important technical lemma, which states the measurability of the *trace* function.

Lemma 5. *$trace : (\mathsf{Execs}, \mathcal{F}_{\mathsf{Execs}}) \to (\mathsf{Traces}, \mathcal{F}_{\mathsf{Traces}})$ is measurable.*

The *trace distribution* corresponding to a probabilistic execution φ_ρ, written as $tdist(\rho)$, is the image measure of φ_ρ under the *trace* function. More formally, $tdist(\rho) : (\mathsf{Traces}_\mathcal{A}, \mathcal{F}_{\mathsf{Traces}_\mathcal{A}}) \to [0, 1]$, is defined as $tdist(\rho)(E) = \varphi_\rho(trace^{-1}(E))$, for any measurable set $E \in \mathcal{F}_{\mathsf{Traces}_\mathcal{A}}$. Note that $trace^{-1}(E) \in \mathcal{F}_{\mathsf{Execs}_\mathcal{A}}$ because *trace* is a measurable function. The *set of trace distributions* of $\mathcal{A}$, $tdists(\mathcal{A})$ is the set of $tdist(\rho)$'s for any task schedule ρ of $\mathcal{A}$.

[2] For $j_i = 0$ the $(i-1)^s t$ and the $i^t h$ intervals are to be concatenated.
[3] **D3** axiom is required here.

4 Hiding and Composition

In this section we define a hiding and parallel composition operations for PDTIOAs. We show that the class of PDTIOAs is closed under these operations. Next we define a simple notion of *external behavior* for PDTIOAs and show that the implementation relation based on this external behavior is compositional.

Definition 6. *Let $\mathcal{A}$ be a PDTIOA and $\mathcal{O}$ be a set of output tasks of $\mathcal{A}$. Let $S = \cup_{T\in\mathcal{O}} T$, that is, S is the set of all actions in the tasks in $\mathcal{O}$. Then,* $\mathsf{ActHide}(\mathcal{A}, S)$ *is defined as PDTIOA $\mathcal{B}$ that is identical to $\mathcal{A}$ except that $O_{\mathcal{B}} = O_{\mathcal{A}} \setminus S$ and $H_{\mathcal{B}} = H_{\mathcal{A}} \cup S$.*

Theorem 3. *If $\mathcal{A}$ be a PDTIOA, $\mathcal{O}$ a set of output tasks of $\mathcal{A}$ and $S = \cup_{T\in\mathcal{O}} T$. Then, $\mathcal{B} = \mathsf{ActHide}(\mathcal{A}, S)$ is also a PDTIOA.*

Definition 7. *Two PDTIOAs $\mathcal{A}_1$ and $\mathcal{A}_2$ are said to be* compatible *if $H_1 \cap A_2 = H_2 \cap A_2 = O_1 \cap O_2 = \emptyset$. The* composition *of two compatible PDTIOAs $\mathcal{A}_1$ and $\mathcal{A}_2$, denoted by $\mathcal{A}_1||\mathcal{A}_2$, is the tuple $\mathcal{A} = ((X, \mathcal{F}_X), \bar{x}, A, \mathcal{R}, \mathcal{D}, \mathcal{T})$, where: (1) $(X, \mathcal{F}_X) = (X_1 \times X_2, \mathcal{F}_{X_1} \otimes \mathcal{F}_{X_2})$, (2) $\bar{x} = (x_1, x_2)$, (3) $A = A_1 \cup A_2, I = (I_1 \cup I_2) \setminus (O_1 \cup O_2), O = O_1 \cup O_2$, and $H = H_1 \cup H_2$. (4) $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2$. (5) $\mathcal{D} \subseteq X \times A \times \mathcal{P}(X, \mathcal{F}_X)$ is the set of triples $((x_1, x_2), a, \mu_1 \otimes \mu_2)$ such that for $i \in \{1, 2\}$ if $a \in A_i$ then $(x_i, a, \mu_i) \in \mathcal{D}_i$, otherwise $\mu_i = \delta_{x_i}$. And (6) $\mathcal{T} = \{\tau \in trajs(X) \mid \tau \downarrow X_i \in \mathcal{T}_i, i \in \{1, 2\}\}$.*

Theorem 4. *If $\mathcal{A}_1$ and $\mathcal{A}_2$ are compatible then $\mathcal{A} = \mathcal{A}_1||\mathcal{A}_2$ is a PDTIOA.*

Proof. We have to verify that $\mathcal{A}$ satisfies all the conditions of Definition 3. Here we present a part of the proof where we show that the set of trajectories $\mathcal{T}$ of $\mathcal{A}$ is maximally continuous (see Appendix A.3 for a complete proof).

Let $max_{\mathcal{T}}(x) = \tau$ for some $x \in X, \tau \in \mathcal{T}$. Let V be an open neighborhood of τ, it suffices to show that there exists an open neighborhood U of x such that $max(U) \subseteq V$. As $\pi_i(V)$ is an open neighborhood of $\tau \downarrow X_i$, and $\mathcal{T}_i$ is maximally continuous, we can find a neighborhood U_i of $(\tau \downarrow X_i).fstate = x \lceil X_i$ such that $max(U_i) \subseteq \pi_i(V)$. Now, $U = U_1 \times U_2$ is an open set in X. Clearly, for every $y \in U$, $max(y) \in V$ which is sufficient for proving $max_{\mathcal{T}}$ is continuous for $\mathcal{A}$.

It is easy to see that $\bar{X}_{\mathcal{T}} = \{x \in X \mid x \lceil X_i \in \bar{X}_{i\mathcal{T}_i}\}$. We check the continuity of $maxstate_{\mathcal{T}} : \bar{X}_{\mathcal{T}} \to X$. Consider any $x \in X$ and let $maxstate_{\mathcal{T}}(x) = x'$, for some $x \in X$. Let V be an open set containing x'. Since $maxstate_{\mathcal{T}_i}$ is continuous for $i \in \{1, 2\}$, we can find open sets U_1, U_2, such that $x \lceil X_i \in U_i$ and $maxstate_{\mathcal{T}_i}(U_i) \subseteq V \lceil X_i$, for $i \in \{1, 2\}$. Therefore, using the same method as above, we define $U = \{x \in X \mid x \lceil X_i \in U_i\}$, and get $maxstate_{\mathcal{T}}(U) \subseteq V$, which is suffices to establish continuity of $maxstate_{\mathcal{T}}$.

The next theorem states an expected property that a projection of an execution of $\mathcal{A}_1||\mathcal{A}_2$ onto the actions and trajectories of $\mathcal{A}_i$ is an execution of $\mathcal{A}_i$, $i \in \{1, 2\}$. The proof of this theory follows immediately from the definition of composition.

Theorem 5. *If $\mathcal{A}_1$ and $\mathcal{A}_2$ are compatible PDTIOAs then α is an execution of $\mathcal{A}_1||\mathcal{A}_2$, then $\pi_i(\alpha)$ is an execution for $\mathcal{A}_i$, for $i \in \{1,2\}$.*

Example 2. Let $\mathcal{A} = \mathsf{ATM}||\ (\mathsf{DATASRC}||\mathsf{TOKENSRC})$ and X be the state space of $\mathcal{A}$. Consider the following execution bases of $\mathcal{A}$, $\Lambda_1 = X[0,t_1]X\{\mathsf{Token}\}X$, $\Lambda_2 = \Lambda_1 X[0,t_2]X\{\mathsf{Data}\}X$, and $\Lambda_3 = \Lambda_2 X[0,t_3]X\{\mathsf{Data},\mathsf{Token},\mathsf{Send}\}X$, where $1 \leq t_1 < \infty$, and $t_2, t_3 > 0$.

$$\varphi_\rho(\mathcal{C}_{\Lambda_2}) = \begin{cases} 1 \text{ if } \ \rho = \{\mathsf{Token}\}\{\mathsf{Data}\}\ldots, \\ 0 \text{ otherwise.} \end{cases}$$
$$\varphi_\rho(\mathcal{C}_{\Lambda_3}) = \begin{cases} 1 - e^{-(\lambda_A+\lambda_D+\lambda_T)t_3} & \text{if } \ \rho = \{\mathsf{Token}\}\{\mathsf{Data}\}T\ldots, \\ 0 & \text{otherwise.} \end{cases}$$

Let $\mathcal{B} = \mathsf{ActHide}(\mathcal{A}, \{\mathsf{Token}\})$. The trace distribution values of $\mathcal{B}$ for $\rho = \{\mathsf{Data}\}\{\mathsf{Send}\}\ldots$, at the points $\Gamma_1 = [0,t_1]\{\mathsf{Data}\}$ and $\Gamma_2 = \Gamma_1[0,t_2]\{\mathsf{Send}\}$ are as follows: $tdist(\rho)(C_{\Gamma_1}) = 1$ and $tdist(\rho)(C_{\Gamma_2}) = 1 - e^{-\lambda_A t_2}$.

Implementation and compositionality. We formulate the external behavior of a $\mathcal{A}$ as a mapping from possible "environments" for $\mathcal{A}$ to sets of trace distributions that can arise when $\mathcal{A}$ is composed with the given environment. The proof of the compositionality theorem that follows is adapted from [2].

Definition 8. *An* environment *for PDTIOA $\mathcal{A}$ is a PDTIOA $\mathcal{E}$ such that $\mathcal{A}$ and $\mathcal{E}$ are compatible and their composition $\mathcal{A}||\mathcal{E}$ is closed. The* external behavior *of a PDTIOA $\mathcal{A}$, written as $extbeh_{\mathcal{A}}$, is defined as a function that maps each environment PDTIOA $\mathcal{E}$ for $\mathcal{A}$ to the set of trace distributions $tdists(\mathcal{A}||\mathcal{E})$.*

Definition 9. *Two PDTIOAs $\mathcal{A}_1$ and $\mathcal{A}_2$ are* comparable *if $E_1 = E_2$. If $\mathcal{A}_1$ and $\mathcal{A}_2$ are comparable then $\mathcal{A}_1$ is said to* implements $\mathcal{A}_2$, *written as $\mathcal{A}_1 \leq \mathcal{A}_2$ if, for every environment PDTIOA $\mathcal{E}$ for both $\mathcal{A}_1$ and $\mathcal{A}_2$, $extbeh_{\mathcal{A}_1}(\mathcal{E}) \subseteq extbeh_{\mathcal{A}_2}(\mathcal{E})$.*

Theorem 6. *Suppose $\mathcal{A}_1, \mathcal{A}_2$ and $\mathcal{B}$ are PDTIOAs, where $\mathcal{A}_1$ and $\mathcal{A}_2$ are comparable and $\mathcal{A}_1 \leq \mathcal{A}_2$. If $\mathcal{B}$ is compatible with $\mathcal{A}_1$ and $\mathcal{A}_2$ then $\mathcal{A}_1||\mathcal{B} \leq \mathcal{A}_2||\mathcal{B}$.*

5 Generalized PDTIOAs and Local Schedulers

PDTIOAs do not allow nondeterministic trajectories nor do they allow choice between enabled actions and non-trivial trajectories. The first restriction is due to the assumption that the set of trajectories of is deterministic and the second one is due to the *Time-action determinism* assumption **D1**. In this section, we briefly discuss one way of relaxing these assumptions by adding *local schedulers*.

Definition 10. *A* Generalized PDTIOA *is a tuple $\mathcal{A} = ((X, \mathcal{F}_X), \bar{x}, A, \mathcal{R}, \mathcal{D}, \mathcal{T})$, where the first five components are the same as in Definition 3. The set $\mathcal{T}$ is not necessarily deterministic and $\mathcal{A}$ does not necessarily satisfy* **D1**.

Thus, from a given state $x \in X$ of a generalized PDTIOA, $\mathcal{A}$ there may be nondeterministic choice of actions that could be performed and also choice of distinct trajectories starting from x. A *local scheduler* for generalized PDTIOA $\mathcal{A}$, is a PDTIOA $S = ((X, \mathcal{F}_X), \bar{x}, A, \mathcal{R}, \mathcal{D}', \mathcal{T}')$ that is identical to $\mathcal{A}$ except that $\mathcal{D}' \subseteq D$ and $\mathcal{T}' \subseteq \mathcal{T}$. A *local scheduler* S satisfies **D1** and has deterministic, maximally continuous trajectories.

A probabilistic system captures the notion of possible ways of resolving the nondeterminism in a generalized PDTIOA. Formally, a *probabilistic-system* is a pair $\mathcal{M} = (\mathcal{A}, \mathcal{S})$, where $\mathcal{A}$ is a generalized PDTIOA and $\mathcal{S}$ is a set of local schedulers for $\mathcal{A}$. An *environment* for $\mathcal{M}$ is any PDTIOA $\mathcal{E}$ such that $\mathcal{A}||\mathcal{E}$ is closed. A probabilistic execution for $\mathcal{M}$ is defined to be any probabilistic execution of S, for any $S \in \mathcal{S}$. For probabilistic system $\mathcal{M} = (\mathcal{A}, \mathcal{S})$, we define the *external behavior* of $\mathcal{M}$ to be the total function $extbeh_{\mathcal{M}}$ that maps each environment PDTIOA $\mathcal{E}$ for $\mathcal{M}$ to the set $\cup_{S' \in \mathcal{S}} tdists(S'||\mathcal{E})$. Thus for each environment, we consider the set of trace distributions that arise from the choices of the local scheduler of $\mathcal{M}$ and the task scheduler ρ. This leads to a notion of implementation of probabilistic systems, similar to that of PDTIOAs.

Definition 11. *Let $\mathcal{M}_1 = (\mathcal{A}_1, \mathcal{S}_1)$ and $\mathcal{M}_2 = (\mathcal{A}_2, \mathcal{S}_2)$ be probabilistic systems such that $\mathcal{A}_1$ and $\mathcal{A}_2$ are comparable generalized PDTIOAs. Then, $\mathcal{M}_1$ is said to* implement *$\mathcal{M}_2$ if for every environment $\mathcal{E}$ of $\mathcal{M}_1$ and $\mathcal{M}_2$, $extbeh_{\mathcal{M}_1}(\mathcal{E}) \subseteq extbeh_{\mathcal{M}_2}(\mathcal{E})$.*

Two probabilistic systems $\mathcal{M}_1 = (\mathcal{A}_1, \mathcal{S}_1)$ and $\mathcal{M}_2 = (\mathcal{A}_2, \mathcal{S}_2)$ are compatible if $\mathcal{A}_1$ and $\mathcal{A}_2$ are compatible, and their composition $\mathcal{M}_1||\mathcal{M}_2$ is defined as $(\mathcal{A}_1||\mathcal{A}_2, \mathcal{S})$, where $\mathcal{S}$ is the set of local schedulers $\{S_1||S_2 \mid S_1 \in \mathcal{S}_1 \text{ and } S_2 \in \mathcal{S}_2\}$. Theorem 7 gives the following sufficient condition for implementation of probabilistic systems: each local schedular for the concrete probabilistic system must always correspond to the same local scheduler for the abstract.

Theorem 7. *If $\mathcal{M}_1 = (\mathcal{A}_1, \mathcal{S}_1)$ and $\mathcal{M}_2 = (\mathcal{A}_2, \mathcal{S}_2)$ are comparable and there exists $f : \mathcal{S}_1 \rightarrow \mathcal{S}_2$, such that for all $S_1 \in \mathcal{S}_1$, S_1 implements $f(S_1)$, then $\mathcal{M}_1$ implements M_2.*

6 Conclusions

We have introduced a compositional framework for modelling and analysis of probabilistic systems over continuous state spaces. The framework supports models with deterministic continuous dynamics, nondeterministic and probabilistic transitions, and nondeterministic stopping times. We have developed the basic mathematical machinery to describe the probabilistic executions and trace distributions of the model.

In the future we will develop new proof techniques and also adapt existing techniques from probability theory, control theory and computer science, for analysis of PDTIOA models. The types of properties that we are particularly interested to investigate are: probabilistic safety and stability properties, both exact and approximate simulation relations for proving implementation. We intend to apply this modelling framework to analyze self-stabilizing timing based distributed algorithms and location aware mobile ad hoc network protocols.

Acknowledgments We greatly benefited from discussing this work with Sanjoy Mitter. We also thank Dilsun Kaynar for giving useful comments on this paper.

References

1. M. Bujorianu and J. Lygeros. General stochastic hybrid systems: Modelling and optimal control. In *IEEE CDC*, Bahamas, December 2004.
2. R. Canetti, L. Cheung, D. Kaynar, M. Liskov, N. Lynch, O. Pereira, and R. Segala. Task-structured probabilistic I/O automata. Technical Report MIT-CSAIL-TR-2006-023, Massachusetts Inst. of Technology, Cambridge, MA, March 2006.
3. S. Cattani, R. Segala, M. Z. Kwiatkowska, and G. Norman. Stochastic transition systems for continuous state spaces and non-determinism. In *FoSSaCS*, 2005.
4. D. Chatterjee and D. Liberzon. Stability analysis of deterministic and stochastic switched systems via a comparison principle and multiple lyapunov functions. *SIAM Journal on Control and Optimization*, 2005.
5. V. Danos, J. Desharnais, F. Laviolette, and P. Panangaden. Bisimulation and cocongruence for probabilistic systems. *Information and Computation, Special issue for selected papers from CMCS04*, 2005.
6. M. H. A. Davis. *Markov Models and Optimization.* Chapman & Hall, 1993.
7. R. M. Dudley. *Real Analysis and Probability.* Wadsworth, Belmont, Calif, 1989.
8. S. Smolka E. Stark, R. Cleaveland. A process-algebraic language for probabilistic I/O automata. In *Proc. CONCUR 03*, *LNCS volume 2761*, Marseille, 2003. Springer.
9. M. Giry. A categorical approach to probability theory. In B. Banaschewski, editor, *Categorical Aspects of Topology and Analysis*, number 915 in Lecture Notes in Mathematics. Springer-Verlag, 1981.
10. M. Hennessy and T. Regan. *A process algebra for timed systems. Information and Computation*, 117:221239, 1995.
11. H. Hermanns. *Interactive Markov Chains : The Quest for Quantified Quality.* Springer Berlin / Heidelberg, 2002.
12. J. P. Hespanha. Stochastic hybrid systems: Application to communication networks. In George J. Pappas Rajeev Alur, editor, *HSCC 2004*, *LNCS volume 2993*, Philadelphia, 2004.
13. D. Kaynar, N. Lynch, R. Segala, and F. Vaandrager. The theory of timed I/O automata. Technical Report MIT/LCS/TR-917a, MIT Laboratory for Computer Sc., 2004. Available at http://theory.lcs.mit.edu/tds/reflist.html.
14. S. Mitra, D. Liberzon, and N. Lynch. Verifying average dwell time by solving optimization problems. In A. Tiwari and J. P. Hespanha, editors, *HSCC 06*, *LNCS volume 3927*, Santa Barbara, 2006. Springer.
15. P. Panangaden. Measure and probability for concurrency theorists. *Theoretical Computer Science*, 253(2), 2001.
16. W. Rudin. *Real & Complex Analysis.* McGraw-Hill, Inc., New York, NY, 3rd edition, 1987.
17. R. Segala. A compositional trace-based semantics for probabilistic automata. In *CONCUR ' 95*, *LNCS volume 962*, Philadelphia, 1995.
18. R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, Laboratory for Computer Sc., Massachusetts Inst. of Technology, June 1995.
19. E. Stark. On behavior equivalence for probabilistic I/O automata and its relationship to probabilistic bisimulation. *Journal of Automata, Languages, and Combinatorics*, 8(2), 2003.

Appendix A: Proofs

A.1 Topologies and semi-rings

Proposition 5. *The collection $\mathscr{B}$ of all trajectory basis elements is a basis for a topology on $\mathcal{T}$.*

Proof. We check that $\mathscr{B}$ satisfies the following two properties:

1. For each $\tau \in \mathcal{T}$, there is at least one basis element B containing τ. Any finite trajectory τ is in the basis element $B_{(q,X)}$ where q is a rational larger than $\sup\{\tau.dom\}$. Any infinite trajectory is in $B_{(\infty,X)}$.
2. If $\tau \in B_{\theta_1} \cap B_{\theta_2}$ then there exists a third basis element B_{θ_3} such that $\tau \in B_{\theta_3} \subseteq B_{\theta_1} \cap B_{\theta_2}$. Let $\theta_1 = (q_1, U_1) \dots (q_m, U_m)$ and $\theta_2 = (r_1, V_1) \dots (r_n, V_n)$. Let $s_1 \dots s_{m+n}$ be the sorted sequence of the $q_1 \dots q_m, r_1 \dots r_n$. We define θ_3 as $(s_1, W_1) \dots (s_{n+m}, W_{n+m})$, where for each $k \in \{1, \dots, m+n\}$, W_k is given by:

$$W_k = U_i \bigcap V_j, \text{where } i = \min\{\omega \mid q_\omega \geq s_k\}$$
$$j = \min\{\omega \mid r_\omega \geq s_k\}$$

   For each k, W_k is an open set in $\mathscr{T}$ and the number s_k is in $\mathbb{Q}_{\geq 0} \cup \{\infty\}$. It can be checked easily that $B_{\theta_3} = B_{\theta_1} \cap B_{\theta_2}$.

Theorem (1). *Let $\mathscr{T}$ be the topology generated by the collection $\mathscr{B}$. $(\mathcal{T}, \mathscr{T})$ is a Hausdorff space.*

Proof. Let $\tau_1, \tau_2 \in \mathcal{T}$ be distinct trajectories. Let B_1, B_2 be basis elements containing τ_1 and τ_2. If $B_1 \cap B_2 = \emptyset$ then we are done, otherwise, consider the two cases:

- **Case 1** $\tau_1.dom = \tau_2.dom$. There exists $r \in \tau_1.dom$, such that $\tau_1(r) \neq \tau_2(r)$. Let U and V be disjoint open neighborhoods of $\tau_1(r)$ and $\tau_2(r)$.

  Claim. There exists $q_1, q_2 \in \mathbb{Q}_{\geq 0}$, $q_1 \leq r < q_2$ so that for all $t \in [q_1, q_2) \cap \tau_1.dom$, $\tau_1(t) \in U$ and $\tau_2(t) \in V$.

  Since $\mathbb{Q}_{\geq 0}$ is dense in $\mathbb{R}_{\geq 0}$, we can choose q_1 and q_2 to be arbitrarily close to r. The claim follows immediately from continuity of τ_1 and τ_2. Suppose, the trajectory base for B_1 is $(r_1, U_1) \dots (r_m, U_m)$ and we assume without loss of generality that $r_i < q_1 < q_2 < r_{i+1}$, for some $i \in \{1, \dots, m\}$. Then, we construct a new basis element B'_1 with the base $(r_1, U_1) \dots (r_i, U_i)(q_1, U_{i+1})$ $(q_2, U)(r_{i+1}, U_{i+1}) \dots (r_m, U_m)$. Similarly, we construct a new basis element B'_2 by inserting (q_2, V) in the trajectory base of B_2. Since U and V are disjoint, we have disjoint open neighborhoods B'_1 and B'_2 for τ_1 and τ_2 respectively.
- **Case 2** $\tau_1.dom \subset \tau_2.dom$. We choose any rational $q > \sup\{\tau_1.dom\}$ and construct a new basis element B'_2 by inserting (q, X) at an appropriate position in the trajectory base for B_2. B_1 and B'_2 are disjoint neighborhoods for τ_1 and τ_2.

Proposition 6. *maxtime is a continuous function.*

Proof. Consider the function $ltime : \mathcal{T} \to \mathbb{R}_{\geq 0}$. We first show that $ltime$ is a continuous function. Suppose $ltime(\tau) = t$, for some $\tau \in \mathcal{T}$ and $t \in \mathbb{R}_{\geq 0} \cup \{\infty\}$. It suffices to show that for any open set I containing t, there exists an open set U_τ containing τ, such that $ltime(U_\tau) \subseteq I$. We select $U_\tau = B_{(q_1,X)(q_2,X)}$ where $q_1 = \inf\{I \cap \mathbb{Q}_{\geq 0}\}, q_2 = \sup\{I \cap (\mathbb{Q}_{\geq 0} \cup \{\infty\})\}$. Clearly, $\tau \in U_\tau$ and for every $\tau' \in U_\tau$, $ltime(\tau') \in I$.

Since max and $ltime$ are both continuous functions, their composition $ltime \circ max = maxtime$ is also continuous.

Lemma (1). *The collection $\mathscr{C}$ of all basic sets of $\mathcal{A}$ is a semi-ring.*

Proof. We check that $\mathscr{C}$ satisfies the three properties of a semi-ring:

1. $\mathscr{C}$ contains the empty set $\emptyset$. Consider a base $\Lambda = X_0R_0X_1A_1X_2 \ldots A_mX_{2m}$ where at least one of the sets in Λ is empty. Then, $C_\Lambda = \emptyset$.
2. If $C_\Lambda, C_\Gamma \in \mathscr{C}$ then there exists a base Δ, such that $C_\Delta = C_\Lambda \cap C_\Gamma$. We can assume without loss of generality that the bases Λ and Γ are of equal length. (If they are not, then we append an appropriately long sequence of $X\mathbb{R}_{\geq 0}XAX$'s to the shorter base to make the two equal. Appending these sets to the base does not change the corresponding basic set.) Let $\Lambda = X_0R_0X_1A_1X_2 \ldots A_mX_{2m}$ and $\Gamma = Y_0T_0Y_1B_1Y_2 \ldots B_mY_{2m}$. Then we define Δ to be the sequence $(X_0 \cap Y_0)(R_0 \cap T_0)(X_1 \cap Y_1)\ (A_1 \cap B_1)(X_2 \cap Y_2) \ldots (A_m \cap B_m)(X_{2m} \cap Y_{2m})$. Now, it can be checked easily that $C_\Delta = C_\Lambda \cap C_\Gamma$.
3. If $C_\Lambda, C_\Gamma \in \mathscr{C}$ and $C_\Lambda \subseteq C_\Gamma$, then there exists a family of basic sets $\{C_i\}_{i \in I}$ such that $C_\Gamma \setminus C_\Lambda = \cup_{i \in I} C_i$ and for all $i, j \in I, C_i \cap C_j = \emptyset$ if $i \neq j$. As in the previous case, we can assume w.l.o.g. that the bases Λ and Γ are of equal length. Let $\Lambda = X_0R_0X_1A_1X_2 \ldots A_mX_{2m}$ and $\Gamma = Y_0S_0Y_1B_1Y_2 \ldots B_mY_{2m}$. Let $I \subseteq \{0, \ldots, 2m\}$, $J \subseteq \{1, \ldots, m\}$, and $K \subseteq \{0, \ldots, m-1\}$ be nonempty sets of indices. We define a collection of bases $\Lambda_{I,J,K} = Z_0T_0Z_1C_1Z_2 \ldots C_mZ_{2m}$ as follows:

$$Z_i = \begin{cases} Y_i \setminus X_i & \text{if } i \in I \\ X_i & \text{otherwise} \end{cases}$$

$$C_j = \begin{cases} B_j \setminus A_j & \text{if } j \in J \\ A_j & \text{otherwise} \end{cases}$$

$$T_k = \begin{cases} S_k \setminus R_k & \text{if } k \in K \\ R_k & \text{otherwise} \end{cases}$$

First, we show that the bases defined above are pairwise disjoint. Consider two bases, $\Lambda_{I,J,K}$ and $\Lambda_{I',J',K'}$. Of the three pairs of index sets, at least one must be a pair of different sets. Say, $J \subset J'$, that is, there exists an index $j \in J'$ such that $j \notin J'$. Then, the j^{th} sets of $\Lambda_{I,J,K}$ and $\Lambda_{I',J',K'}$ are disjoint. It follows that the $C_{\Lambda_{I,J,K}} \cap C_{\Lambda_{I',J',K'}} = \emptyset$. Next, we check that any $\alpha \in C_\Lambda \setminus C_\Gamma$ is in one of the basic sets constructed above. Let $\alpha = \tau_0 a_1 \ldots a_m \beta$. Then there must exist index sets $I_\alpha \subseteq \{0, \ldots, 2m\}$, $J_\alpha \subseteq \{1, \ldots, m\}$ and $K_\alpha \subseteq \{0, \ldots, m-1\}$ such that $\tau_{\frac{i}{2}}.fstate \in Y_i \setminus X_i$ for all even $i \in I_\alpha$, $\tau_{\frac{i-1}{2}}.lstate \in Y_i \setminus X_i$ for all odd $i \in I_\alpha$, $a_j \in B_j \setminus A_j$ for all $j \in J_\alpha$, and $\tau_k.ltime \in S_k \setminus R_k$ for all $k \in K_\alpha$.

A.2 Measurability of trace function

Definition 12. *Let $\mathscr{J}$ be the set of intervals in $\mathbb{R}_{\geq 0}$. For any $k \in \mathbb{N}$ the function $rcuts_k : \mathscr{J} \to \mathscr{J}^{2^k}$ is defined recursively as:*

$$rcut_0((a,b)) = \{(a,b)\} \text{ and}$$

$$rcut_k((a,b)) = \{rcut_{k-1}((a,q]), rcut_{k-1}([q,b))\}, \quad \text{where } q \in \mathbb{Q}, \frac{2a+b}{3} \leq q \leq \frac{2b+a}{3}.$$

*For closed and half open intervals the function is defined analogously. Given $(a,b) \in \mathscr{J}$, a k-*partition *is a sequence of intervals $(a,q_1],[q_1,q_2],\ldots,[q_n,b)$ that partitions (a,b) such that every interval in the sequence is in $\bigcup_{i=1}^{k} rcut_k((a,b))$. We denote the collection of all k-partitions of (a,b) by $S_k((a,b))$.*

Proposition (4). *Given an interval I and a real number $r \in I$, there exists a k-partition of I with an endpoint that coincides with r, as $k \to \infty$.*

Proof. Let r_k be the nearest left endpoint of any interval containing r in the set of all k-partitions $S_k(I)$. From the definition of $rcut_k$ it follows that $r - r_k \leq |I|(\frac{2}{3})^k$. Thus, $r_k \to r$, as $k \to \infty$.

A.3 Composition

Theorem (4). *If $\mathcal{A}_1$ and $\mathcal{A}_2$ are compatible PDTIOAs then $\mathcal{A} = \mathcal{A}_1||\mathcal{A}_2$ is a PDTIOA.*

Proof. We show that $\mathcal{A}$ satisfies all the properties of a PDTIOA:

1. **M1** For all $a \in A$, set of states at which a is enabled is a measurable set. If $a \in A_1$ then we know that the set of states $Y_1 \subseteq X_1$ in which a is enabled is in $\mathcal{F}_{X_1}$. The set $Y_1 \times X_2 \in \mathcal{F}_X$, and it is precisely the set of states in which a is enabled.
   M2 Follows from an exactly similar argument.
2. $\mathcal{T}$ is deterministic. Consider two distinct trajectories $\tau, \zeta \in \mathcal{T}(x)$, for some $x \in X$ with $\tau.dol \subseteq \zeta.dom$. Since $(\tau \downarrow X_i), (\zeta \downarrow X_i) \in \mathcal{T}_i(x \lceil X_i)$, for $i \in \{1,2\}$ and $\mathcal{T}_i$ is a deterministic set of trajectories for X_i, it follows that $(\tau \downarrow X_i) \leq (\zeta \downarrow X_i)$. Combining this result for 1 and 2, we get $\tau \leq \zeta$.
3. $\mathcal{T}$ is closed under prefix, suffix, concatenation and for every $x \in X$, $\wp(x) \in \mathcal{T}$. These follow immediately from the definition of $\mathcal{T}$.
4. $\mathcal{T}$ is maximally continuous. Let $max_{\mathcal{T}}(x) = \tau$ for some $x \in X, \tau \in \mathcal{T}$. Let V be an open neighborhood of τ, it suffices to show that there exists an open neighborhood U of x such that $max(U) \subseteq V$. As $\pi_i(V)$ is an open neighborhood of $\tau \downarrow X_i$, and $\mathcal{T}_i$ is maximally continuous, we can find a neighborhood U_i of $(\tau \downarrow X_i).fstate = x \lceil X_i$ such that $max(U_i) \subseteq \pi_i(V)$. Now, $U = U_1 \times U_2$ is an open set in X. Clearly, for every $y \in U$, $max(y) \in V$ which is sufficient for proving $max_{\mathcal{T}}$ is continuous for $\mathcal{A}$.
   It is easy to see that $\bar{X}_{\mathcal{T}} = \{x \in X \mid x \lceil X_i \in \bar{X}_{i\mathcal{T}_i}\}$. We check the continuity of $maxstate_{\mathcal{T}} : \bar{X}_{\mathcal{T}} \to X$. Consider any $x \in X$ and let $maxstate_{\mathcal{T}}(x) = x'$,

for some $x \in X$. Let V be an open set containing x'. Since $maxstate_{\mathcal{T}_i}$ is continuous for $i \in \{1,2\}$, we can find open sets U_1, U_2, such that $x \lceil X_i \in U_i$ and $maxstate_{\mathcal{T}_i}(U_i) \subseteq V \lceil X_i$, for $i \in \{1,2\}$. Therefore, using the same method as above, we define $U = \{x \in X \mid x \lceil X_i \in U_i\}$, and get $maxstate_{\mathcal{T}}(U) \subseteq V$, which is suffices to establish continuity of $maxstate_{\mathcal{T}}$.

5. **D0, D2** and **D3** follow from the definition of composition.
6. **D1** $\mathcal{A}$ is time-local action deterministic. Suppose some local action $a \in L$ is enabled at state x. Let us assume without loss of generality that $a \in L_1$ and $a \notin L_2$. Then a is enabled at $x \lceil X_1$ and since $\mathcal{A}$ satisfies **D1** it follows that there does not exist any non-point trajectory in $\mathcal{T}_1$ (and therefore in $\mathcal{T}$) that starts from $x \lceil X_1$.
   Likewise, it is easy to check that if there exists a non-point trajectory starting from x, then no local action is enabled at either $x \lceil X_1$nor $x \lceil X_2$.

Theorem (6). *Suppose $\mathcal{A}_1, \mathcal{A}_2$ and $\mathcal{B}$ are PDTIOAs, where $\mathcal{A}_1$ and $\mathcal{A}_2$ are comparable and $\mathcal{A}_1 \leq \mathcal{A}_2$. If $\mathcal{B}$ is compatible with each of $\mathcal{A}_1$ and $\mathcal{A}_2$ then $\mathcal{A}_1||\mathcal{B} \leq \mathcal{A}_2||\mathcal{B}$.*

Proof. Let $\mathcal{E}$ be an environment PDTIOA for both $\mathcal{A}_1||\mathcal{B}$ and $\mathcal{A}_2||\mathcal{B}$. Consider a task schedule ρ_1 for the composed PDTIOA $(\mathcal{A}_1||\mathcal{B})||\mathcal{E}$. Let $\eta = tdist(\rho_1)$ be the trace distribution of $(\mathcal{A}_1||\mathcal{B})||\mathcal{E}$ generated by ρ_1. It suffices to show that η is also a trace distribution of $(\mathcal{A}_2||\mathcal{B})||\mathcal{E}$, generated by some task schedule.

As ρ_1 is a task schedule for $\mathcal{A}_1||(\mathcal{B}||\mathcal{E})$ it generates the same trace distribution η for PDTIOA $\mathcal{A}$ composed with the environment $\mathcal{B}||\mathcal{E}$. Further $\mathcal{B}||\mathcal{E}$ is also a closing environment for $\mathcal{A}_2$ because $\mathcal{A}_1$ and $\mathcal{A}_2$ are compatible. As $A_1 \leq A_2$, there exists a task schedule ρ_2 for $\mathcal{A}_2||(\mathcal{B}||\mathcal{E})$ that generates the trace distribution η. It follows that ρ_2 is a task schedule for $(A_2||\mathcal{B})||\mathcal{E}$ that produces the trace distribution η.

Appendix B: Examples

B.1 Leaking bucket: Source and Token

The code in Figure 2 specify the PDTIOAs representing the source of data packets and the token, for the ATM automaton. Data packets and tokens are generated with exponentially distributed inter-arrival times, with rates λ_D and λ_T respectively. As in the model presented in [11], after each action Data, Token, or Send, all the clocks in the system are reset and new delays are fixed.

B.2 Periodically sending process

Example 3. Consider the simple PDTIOA PeriodicSend shown in Figure 3. The state space X of PeriodicSend is the product $\mathbb{R}^2_{\geq 0} \times \{\mathsf{true}, \mathsf{false}\}^2$; the set of actions A is $\{\mathsf{tick}, \mathsf{fail}\} \cup \bigcup_{i=1}^{n}\{\mathsf{send}_i\}$. Typical executions of PeriodicSend are as follows: until a fail action is received, PeriodicSend triggers alternating send_i and tick actions, with interleaving trajectories. The length of the trajectories are

```
DATASRC($\lambda_D$) where $\lambda_D \in \mathbb{R}_+$
Variables:
  $clock_D : \mathbb{R}_+$ initially $0$
  $delay_D : \mathbb{R}_+$ initially $1$

Actions:
  output Data($m$) $m \in \{0,1\}$
  input Token, Send($m$) $m \in \{0,1\}$

Transitions:
  Send($m$) eff $Reset_D$

  Data($m$)
  pre $clock_D = delay_D$
  eff $Reset_D$

  Token  eff $Reset_D$

Trajectories:
  Trajdef $normal$
  invariant $clock_D \leq delay_D$
  evolve $d(clock_D) = 1$

Tasks:  {Data}

Subroutine $Reset_D$:
  $delay_D$ := choose $Exp(\lambda_D)$;
  $clock_D := 0$
```

```
TOKENSRC($\lambda_T$) where $\lambda_T \in \mathbb{R}_+$
Variables:
  $clock_T : \mathbb{R}_+$ initially $0$
  $delay_T : \mathbb{R}_+$ initially $1$

Actions:
  output Token
  input Data($m$), Send($m$) $m \in \{0,1\}$

Transitions:
  Send($m$) eff $Reset_T$

  Data($m$) eff $Reset_T$

  Token
  pre $clock_T = delay_T$
  eff $Reset_T$

Trajectories:
  Trajdef $normal$
  invariant $clock_T \leq delay_T$
  evolve $d(clock_T) = 1$

Tasks:  {Token}

Subroutine $Reset_T$:
  $delay_T$ := choose $Exp(\lambda_T)$;
  $clock_T := 0$
```

Fig. 2. Automaton DATASRC and TOKENSRC.

chosen uniformly at random over $[a, b]$. PeriodicSend is nondeterministic because if send_j is enabled at some state then send_i is also enabled at the same state, for every $i, j \in \{1, \ldots, n\}$. Note that each send_i belongs to a different task. A typical execution of PeriodicSend $(n > 4)$ and the corresponding trace looks like this:

$$\alpha = \tau_1 \ \mathsf{tick} \ \tau_2 \ \mathsf{send}_i \ \tau_3 \ \mathsf{tick} \ \tau_4 \ \mathsf{send}_k \ \tau_5 \ldots, \text{and}$$
$$trace(\alpha) = (\tau_1.ltime + \tau_2.ltime) \ \mathsf{send}_i \ (\tau_3.ltime + \tau_4.ltime) \ \mathsf{send}_k \ \tau_5.ltime \ldots$$

where $\tau_j.dom \subseteq [0, b]$ for each j and $i, k \in \{1, \ldots, n\}$.

Example 4. Consider a version of the PeriodicSend automaton of Figure 3 with the fail action and the $failed$ variable removed. The automaton thus obtained is a closed PDTIOA. Let us call it $\overline{\mathsf{PeriodicSend}}$. Let $\Lambda = X_0 R_0 X_1 A_1 X_2 \ldots X_{2m-1} A_m X_{2m}$ be a base for $\overline{\mathsf{PeriodicSend}}$ with:

$$X_i = (\mathbb{R}_+)^2 \times \{\mathsf{true}, \mathsf{false}\}^2, \quad \text{for each } i \in \{0, \ldots, 2m\},$$
$$R_i = [0, b_i), \quad \text{for each } i \in \{0, \ldots, m-1\}, \text{where } a \leq b_1 \leq b,$$
$$A_i \subseteq \{\mathsf{tick}, \mathsf{send}_1, \ldots, \mathsf{send}_n\}, \quad \text{for each } i \in \{1, \ldots, m\}.$$

Let $\rho = T_1 T_2 \ldots T_l$ be a task schedule for $\overline{\mathsf{PeriodicSend}}$, and $p_i = \min(1, \frac{b_i - a}{b - a})$, for each $i \in \{0, \ldots m-1\}$. It can be checked that the probabilistic execution of

```
PeriodicSend$(a, b, n)$ where $a, b \in \mathbb{R}_+$, $n \in \mathbb{N}$
Variables:
  $clock$: $\mathbb{R}_+$ := 0
  $u$ :$\mathbb{R}_+$ := $a$
  $flag$, $failed$: $bool$ := false

Actions:
  output send$_i$ $i \in \{1,2, \ldots, n\}$
  input fail, internal tick

Transitions:
  send$_i$
  pre $(\neg\, failed) \wedge flag \wedge clock = u$
  eff $clock$ := 0, $flag$ = true,
    $u$ := choose uniformly $[a, b]$

  tick
  pre $(\neg\, failed) \wedge (\neg\, flag) \wedge clock = u$
  eff $clock$ := 0, $flag$ = false,
    $u$ := choose uniformly $[a, b]$

  fail
  eff $failed$ := true

Trajectories:
  Trajdef $normal$
  stop when $clock = u$
  evolve $d(clock) = 1$

Tasks:
  {tick} and {send$_i$} for $i \in \{1, 2, \ldots, n\}$
```

Fig. 3. Automaton PeriodicSend: u_A is chosen uniformly at random over $[a, b]$.

$\overline{\mathsf{PeriodicSend}}$ corresponding to ρ, is given by:

$$\varphi_\rho(\mathcal{C}_\Lambda) = \begin{cases} \prod_{i=0}^{m-1} p_i & \text{if } l \geq m \text{ and } A_i \cap T_{m-i+1} \neq \emptyset, \text{ for each } i \in \{1, \ldots, m\}, \\ 0 & \text{otherwise.} \end{cases}$$