Probabilistic Language Formalism for Stochastic Discrete Event Systems [1] [2]

Vijay K. Garg
Department of Electrical and Computer Engineering
University of Texas at Austin
Austin, Texas 78712-1084

Ratnesh Kumar
Department of Electrical Engineering
University of Kentucky
Lexington, KY 40506-0046

Steven I. Marcus
Department of Electrical Engineering and
Institute for Systems Research
University of Maryland at College Park
College Park, MD 20742

[1] Preliminary versions of this paper appeared as [7, 6].

[2] This research was supported in part by the Center for Robotics and Manufacturing, University of Kentucky, in part by the National Science Foundation under Grants NSF-ECS-9409712, NSF-ECS-9414780, NSFD-CDR-8803012, and NSF-EEC-9402384, in part by the Air Force Office of Scientific Research (AFOSR) under Contract F49620-92-J-0045, and in part by a TRW faculty assistantship award.

Abstract

The formalism of probabilistic languages has been introduced for modeling the qualitative behavior of stochastic discrete event systems. A probabilistic language is a unit interval valued map over the set of traces of the system satisfying certain consistency constraints. Regular language operators such as choice, concatenation, and Kleene-closure have been defined in the setting of probabilistic languages to allow modeling of complex systems in terms of simpler ones. The set of probabilistic languages is closed under such operators thus forming an algebra. It also is a complete partial order under a natural ordering in which the operators are continuous. Hence recursive equations can be solved in this algebra. This is alternatively derived by using contraction mapping theorem on the set of probabilistic languages which is shown to be a complete metric space. The notion of regularity, i.e., finiteness of automata representation of probabilistic languages has been defined and shown that regularity is preserved under choice, concatenation, and Kleene-closure. We show that this formalism is also useful in describing system performances such as completion time, reliability, etc. and present properties to aide their computation.

Keywords: Discrete Event Systems, Stochastic Systems, Automata, Languages, Regularity, Completion time, Reliability

1 Introduction

Discrete event systems are systems which involve quantities that are discrete and which change when events occur in the system. Most prior work on characterizing qualitative behavior of such *event driven* systems has been restricted to *non-stochastic* systems where the emphasis is on possibilities rather than on probabilities. The theory of *time driven stochastic* systems on the other hand has been well developed and understood. In this paper we introduce the formalism of probabilistic languages to describe the behavior of stochastic discrete event systems. Such systems are also represented as nondeterministic automata with probabilities associated with transitions.

Consider for example a machine which operates in one of the three possible states—idle, working, and broken. In the idle state, the machine state changes to the working state with probability one when the "commence" event occurs. In the working state, the machine state changes to the broken state with probability, say, p when the "breakdown" event occurs, whereas it changes to the idle state with the remainder probability $1-p$ when the "completion" event is executed. Finally, in the broken state, the machine state changes to the idle state with probability one when the "repair" event is executed.

Such a stochastic discrete event system can clearly be represented by an automaton with probabilities associated with transitions such that the probabilities of all transitions from any state add up to at most one. When the probabilities do not add up to exactly one, this means the systems terminates with the remainder probability. Thus by modeling systems by "substochastic automata" we model the possibility of termination, which also allows us to model concatenation or sequential operation of two or more systems. The substochastic automata model can be converted to an equivalent stochastic automata model by introducing an auxiliary "termination" event and assigning appropriate probability to each "termination" transition so that the transition probabilities from any state add up to exactly one. Thus this model is similar to the well known Markov chain models [2]. The difference lies in the fact that unlike the Markov chain models the transition event labels are emphasized so that we can study the event traces and their occurrence probabilities. The work on stochastic Petri nets [13] also follows the work of Markov chains and is thus different from the work presented here.

Our model also differs from Rabin's probabilistic automata model [17] where the probabilities on transitions on *each* event (rather than *all* events) from any state add up to one. Thus in Rabin's model system changes its state on *each* event with probability one. However in our model the cumulative probability of state change over *all* events is at most one. Rabin's model is weaker in the sense that less information about state change on an event occurrence is available. The motivation for Rabin's work was to introduce the notion of "cut-languages". A cut-language is the set of accepted event traces whose occurrence probability exceeds a given cut value. Our motivation here is to study the probabilistic languages associated with models bearing similarity to Markov chain models. Thus a twofold difference lies between Rabin's probabilistic automata and the model presented here. Mortazavian [15] considered the supervisory control problem for systems modeled as Rabin's probabilistic automata. Thus our work is also distinct from that of Mortzavian. There have been other attempts to generalize deterministic automata to the probabilistic setting [16, 4] which also follow the lines of Rabin's model.

As is defined below a probabilistic language associates a probability (a value in the unit interval) to each finite length trace of the system with the following two constraints on the probabilities: (i) the probability of the zero length trace is one, and (ii) the probability of any trace is at the least the cumulative probability of all its extensions. In [8] a probability measure on the set of traces is defined to be a probability map satisfying the first constraint above and a stronger second constraint that requires that the probability of each trace be equal the cumulative probability of all its extensions. By relaxing this constraint we are able to model the possibility of termination, which also allows us to define concatenation or sequential operation of two or more stochastic discrete event systems.

A probabilistic language can be viewed a *formal power series* [20]—a map over the set of traces, but with the additional two constraints described above. There are many important differences in our work from the classical work on formal series. Our definition of concatenation (product) and therefore of concatenation-closure is quite different from that of classical definition of product. The classical definition of product, called Cauchy product, is the same as the convolution operator used in this paper. Our definition of product is more useful in modeling concatenation or sequential operation of two or more systems.

Another early attempt to associate probabilities with traces was done by researchers in fuzzy set theory [12]. A fuzzy language is a fuzzy set of event traces, so that each trace has a membership grade in the unit interval $[0, 1]$. Our definition of probabilistic languages allows only those membership grades which satisfy certain consistency constraints. A consequence of these constraints is that the membership grade of a trace can be viewed as the probability that the system executes that trace. This results in a richer theory for stochastic discrete event system modeled by probabilistic languages.

We define regular language operators such as choice, concatenation, and Kleene-closure over the domain of probabilistic languages to allow modeling of complex systems in terms of simpler ones. We show that the set of probabilistic languages is closed under such operators, thus forming an algebra. We show that this set is a well structured poset—a complete partial order—under a natural partial order in which the operators are continuous. Hence recursive equations can be solved in this algebra. We also establish this result alternatively by defining a metric on the set of probabilistic languages in which the set is complete and the operators are continuous and contracting. Consequently, the contraction mapping theorem applies yielding existence of unique fixed points of the operators. We characterize the regularity of probabilistic languages, i.e., finiteness of their automaton representation and show that the operators preserve regularity. Thus, probabilistic languages forms a suitable formalism for modeling and analysis of stochastic discrete event systems.

We show that probabilistic language formalism is also useful in describing system performances such as average completion time and reliability. We present analytical techniques to aide in computation of "bilinear" performance functions by showing that they enjoy several compositional properties so that performances of complex systems can be easily obtained in terms of those of their simpler constituent sub-systems. Computation of performance functions by recursion over regular operations was first investigated in [14]. Our results extend this technique to recursion over probabilistic regular operations. The formalism of probabilistic languages can also be used for studying supervisory control problems [18, 11] of stochastic discrete event systems. The concluding section discusses some ways in which control problems can be studied in this formalism. More detailed examples and algorithms are subject of work under progress.

The rest of the paper is organized as follows. Section 2 describes the notation in this paper. Section 3 defines probabilistic languages and introduces a partial order on the class of probabilistic languages making it a complete partial order. Section 4 describes various operators defined for probabilistic languages and studies their properties. Section 5 defines probabilistic automata and relates them to probabilistic languages that are regular. Section 6 defines system performances and describes compositional methods to compute them. Section 7 concludes the work presented here and discusses some control problems that can be studied in this formalism. Appendix A gives a metric in which the set of probabilistic languages is complete and the operators are continuous and contracting.

2 Notation and Preliminaries

We use Σ to denote the universe of events. Σ^* denotes the set of all finite length sequences of events from Σ, including the zero length trace ϵ. A member of Σ^* is called a trace, and a subset of it is called a language. Given traces s and t, we use $s \leq t$ to denote that s is a prefix of t, in which case the notation $s^{-1}t$ is used to denote the suffix of t obtained by removing the prefix s, i.e., $t = ss^{-1}t$. A language is called prefix closed if it contains all prefixes of all of its traces. Thus a prefix closed non-empty language always contains ϵ.

A nondeterministic automaton G over the event set Σ is a triple $G := (X, x_0, \delta)$, where X is the set of states, $x_0 \in X$ is the initial state, and $\delta : X \times \Sigma \to 2^X$ is the partial nondeterministic transition function which gives the set of possible resulting states when a certain event in a given state occurs. For more details on languages and automata refer to [9].

Given a set X, a partial order on X, denoted $\preceq$, is a binary relation that is reflexive, anti-symmetric, and transitive. The pair $(X, \preceq)$ is called a partially ordered set or a poset. Given a poset $(X, \preceq)$ and a pair of elements $x, y \in X$, their infimum and supremum whenever defined are unique and are denoted by $x \sqcap y$ and $x \sqcup y$, respectively. The poset $(X, \preceq)$ is called an inf semi-lattice (respectively, a sup semi-lattice) whenever the infimum (respectively, supremum) is defined for any pair of poset elements. It is called a lattice if it is both inf as well as sup semi-lattice. $x \in X$ is called the bottom (respectively, the top) element if $x \leq y$ (respectively, $y \leq x$) for each $y \in X$.

A chain in a poset is a monotonically increasing sequence of poset elements. I.e., $\{x_i\}_{i \geq 0}$, where $x_i \in X$ for each i, is called a chain if $x_i \preceq x_j$ whenever $i \leq j$. A poset $(X, \preceq)$ is called a complete partial order (cpo) if it contains the bottom element of X, and the supremum element of each chain.

Given a poset $(X, \preceq)$, a function $f : X \to X$ is called monotone if the partial ordering is preserved under its transformation, i.e., $x \preceq y$ implies $f(x) \preceq f(y)$. The function is called continuous if it commutes with the supremum operation taken over a chain, i.e., given a chain $\{x_i\}$, continuity of f requires that $f(\sqcup_i x_i) = \sqcup_i f(x_i)$. Note that for continuity of f to be defined, X must be a cpo (so that $\sqcup x_i$ is defined for a chain $\{x_i\}$). It is easy to show that every continuous function is also monotone.

$x \in X$ is called a fixed point of $f : X \to X$ if $f(x) = x$. It is known that every continuous function defined over a cpo $(X, \preceq)$ possesses a infimum fixed point which is given by $\sqcup_{i \geq 0} f^i(\perp)$, where $\perp$ denotes the bottom element of X, f^i denotes the i-fold application of f, and f^0 is the identity function. Thus recursive equations of the type $f(x) = x$ can be solved whenever f is continuous and is defined on a cpo. We use this fact to define recursive equations in the domain of probabilistic languages. For further reading on posets and functions defined on posets refer to [3].

Given a set Ω, a σ-algebra on Ω is a set of subsets of Ω such that it is closed under complementation and countable union, and contains Ω. The set of σ-algebras on Ω is closed under arbitrary intersection. So given a collection of subsets of Ω, there exists the smallest σ-algebra containing the given subsets. This is called the σ-algebra generated by the given subsets. A measurable space is a pair $(\Omega, \mathcal{F})$, where Ω is an arbitrary set, called a *sample space*, and $\mathcal{F}$ is a σ-algebra on Ω, each member of which is called a *measurable set*. Given a measurable space $(\Omega, \mathcal{F})$, a probability measure P on it is map $P : \mathcal{F} \to [0, 1]$ that maps (i) Ω to one, and (ii) satisfies σ-additivity, i.e., probability measure of countable union of pairwise disjoint sets in $\mathcal{F}$ equals the sum of their individual probability measures. The triple $(\Omega, \mathcal{F}, P)$ is called a *probability space*. For further detail on measurable spaces and probability measures refer to [21].

3 Probabilistic Languages

A language can be viewed as a binary valued map over the set of traces in Σ^* which assigns a unit value to a trace if and only if the trace belongs to the language. If this language represents the qualitative behavior of a certain discrete event system, then it must be non-empty and prefix closed, i.e., the zero length trace must be mapped to one, and whenever a trace is mapped to one any of its prefix must also be mapped to one.

We view the probabilistic language of a stochastic discrete event system in a similar manner which assigns a probability measure (a value in the unit interval) to each trace in Σ^* with the interpretation that this value determines its probability of its occurrence. Since a zero length trace is always possible in a system, its probability measure must be one. Also, for a trace to occur, all its prefixes must occur first. So the cumulative probability of all traces sharing a common prefix should not exceed the probability of the prefix itself. This captured by properties P1 and P2 below.

In order to represent the qualitative behavior of a stochastic discrete event system, we first describe the underlying measurable space. For mathematical convenience we explicitly represent the occurrence of termination by augmenting the event set with the "termination event", denoted σ_Δ. Then the sample space for the behavior of a stochastic discrete event system is given by the set of all finite length traces possibly followed by the termination event, i.e., $\Omega = \Sigma^*(\sigma_\Delta + \epsilon) = \Sigma^*\sigma_\Delta \cup \Sigma^*$. Since we are interested in the probability of "occurrence of a trace $s \in \Omega$"—which occurs whenever any trace containing s as its prefix executes, we let the set of all traces having s as a prefix be a measurable set. We use $< s > := \{st \mid st \in \Omega\}$ (and simply s when there is no contextual confusion) to denote this measurable set. Then the set of all measurable sets $\mathcal{F}$ is the σ-algebra generated by $\{< s > \mid s \in \Omega\}$.

Definition 1 Consider the measurable space $(\Omega, \mathcal{F})$, where $\Omega = \Sigma^*(\Sigma_\Delta + \epsilon)$ and $\mathcal{F}$ is the σ-algebra generated by $\{< s > \mid s \in \Omega\}$. Then a *probabilistic language (p-language)* L is a probability measure on the measurable space $(\Omega, \mathcal{F})$, i.e., it is a map $L : \mathcal{F} \to [0, 1]$ that maps Ω to one and satisfies the property of σ-additivity.

Remark 1 Since (i) the probability measure L on the σ-algebra $\mathcal{F}$ is uniquely determined by its value on the set $\{< s > \mid s \in \Omega\} \subseteq \mathcal{F}$ which generates the σ-algebra $\mathcal{F}$, and (ii) the set $\{< s > \mid s \in \Omega\}$ has the obvious one to one correspondence with the set Ω, we can view L to be a unit interval valued map on Ω. With this correspondence in mind for each $s \in \Omega$, we use $L(s)$ to represent the probability measure $L(< s >)$, and refer to it as the probability of occurrence of trace s.

Next, due to the property of σ-additivity of L, we have

$$\forall s \in \Sigma^* : L(s\sigma_\Delta) = L(s) - \sum_{\sigma \in \Sigma} L(s\sigma), \tag{1}$$

where $L(s\sigma_\Delta)$ represents the probability of termination following s, $L(s)$ represents the probability of occurrence of s, and $\sum_{\sigma\in\Sigma} L(s\sigma)$ represents the probability of continuous operation beyond s. Note that in Equation (1) $s \in \Sigma^*$, so $s\sigma_\Delta \in \Sigma^*\sigma_\Delta$ and $s\sigma \in \Sigma^*$ for each $\sigma \in \Sigma$. Thus the value of L on $\Sigma^*\sigma_\Delta$ is uniquely determined by its vale on Σ^*. Hence it suffices to view L as a unit interval valued map on Σ^* (rather than on its superset Ω), which we do from here on.

For future notational convenience we define

$$\forall s \in \Sigma^* : \Delta(L)(s) := L(s\sigma_\Delta); \quad \Lambda(L)(s) := \sum_{\sigma \in \Sigma} L(s\sigma)$$

to be the probability of termination and probability of continuous operation, respectively, following the execution of trace s. Then it is clear that for any L, $\Delta(L)$ and $\Lambda(L)$ are both unit interval valued maps on Σ^*. The Equation (1) can be concisely written as: $\Delta(L) = L - \Lambda(L)$.

Remark 2 Consider the measurable space $(\Omega, \mathcal{F})$ as given in Definition 1, and a p-language $L : \mathcal{F} \to [0, 1]$. Then it is easy to see that the following two properties hold when L is viewed as a unit interval valued map on Σ^*:

P1: $L(\epsilon) = 1$
P2: $L - \Lambda(L) \geq 0$

P1 follows from the fact that $L(\epsilon) = L(<\epsilon>) = L(\Omega) = 1$, whereas P2 follows from the fact that for each $s \in \Sigma^*$, $L(s) - \Lambda(L)(s) = L(s\sigma_\Delta) \geq 0$.

Conversely, given a map $L : \Sigma^* \to [0,1]$ satisfying P1 and P2 it can be extended in the following manner to obtain a probability measure on $(\Omega, \mathcal{F})$: First Equation (1) is used to extend L from Σ^* to Ω. Next for any $s \in \Omega$, the probability measure of the set $<s> \in \mathcal{F}$ is simply given by $L(s)$. Finally, the probability measure for the collection of sets $\{<s>| \, s \in \Omega\}$ to obtain the probability measure of any set in $\mathcal{F}$ by applying the property of σ-additivity.

The above observation motivates the following the simplified definition of a p-language describing the qualitative behavior of a stochastic discrete event system.

Definition 2 A p-language L is a unit interval valued map on Σ^* satisfying P1 and P2. We use $\mathcal{L}$ to denote the set of all p-languages.

Since the qualitative behavior of a stochastic discrete event system is completely described by its p-language, the terms "system" and "p-language" are used below synonymously.

Remark 3 A consequence of P2 is that $L(s) \geq L(t)$ whenever s is a prefix of t. Also note that for any trace s, $L(s) \leq 1$ can also be derived from P1 and P2. However, we prefer to keep it as part of the definition for simplicity.

Example 1 Consider a system that deadlocks initially. We use *nil* p-language I to represent its behavior. Then $I(\epsilon) = 1$, and $I(s) = 0$ for $s \neq \epsilon$. It is easy to verify that P1 and P2 hold for I; and $\Delta(I) = I$.

Consider a Bernoulli process where each experiment has two outcomes a and b with probabilities p and $(1-p)$, respectively. Here the event set $\Sigma = \{a, b\}$, and the associated p-language L is given by:

$$\forall s \in \Sigma^* : L(s) = p^{\#(a,s)}(1-p)^{\#(b,s)},$$

where $\#(a, s)$ represents the number of occurrences of a in the trace s. It is clear that P1 and P2 hold for L; and $\Delta(L) = 0$.

Note that the outcomes corresponding to terminations after distinct traces are mutually exclusive. Hence the cumulative probability of termination of a system can be obtained by adding the individual probabilities over all possible traces: $\sum_s \Delta(L)(s)$. For the system with nil language this cumulative probability equals one which implies that it is a terminating system. On the other hand, for the Bernoulli process this cumulative probability is zero implying that this system does not terminate. In the following theorem we establish that this cumulative probability of termination is bounded above by one (a system is not necessarily guaranteed to terminate), and it equals one if and only if the probability of traces of arbitrary length converges to zero.

Theorem 1 Consider a p-language L. Then

1. $\sum_s \Delta(L)(s) \leq 1$
2. $[\sum_s \Delta(L)(s) = 1 \iff \lim_{k\to\infty} \sum_{|t|=k} L(t) = 0$

Proof: 1. Define $S(n) := \sum_{|s|\leq n} \Delta(L)(s)$ to be the probability of termination in at most n steps of execution, so that $\sum_s \Delta(L)(s) = \lim_{n\to\infty} S(n)$. Then since $S(n)$ is monotonically increasing, it suffices to show that it is bounded above by one. (This implies that its limiting value is also bounded above by one.)

We first show using induction on n that $S(n) = 1 - \sum_{|s|=n+1} L(s)$. Clearly this holds for $n = 0$, since in that case $S(n) = C(\epsilon)$ and $1 - \sum_{|s|=1} L(s) = L(\epsilon) - \sum_\sigma L(\sigma) = C(\epsilon)$. This establishes the base step. For the induction step we have

$$\begin{aligned}
S(n+1) &= S(n) + \sum_{|s|=n+1} \Delta(L)(s) \\
&\quad \{\text{from induction hypothesis and definition of } \Delta\} \\
&= [1 - \sum_{|s|=n+1} L(s)] + \sum_{|s|=n+1} [L(s) - \sum_\sigma L(s\sigma)] \\
&= 1 - \sum_{|s|=n+1} \sum_\sigma L(s\sigma) \\
&= 1 - \sum_{|s|=n+2} L(s),
\end{aligned}$$

This establishes the induction step.

Now since $S(n) = 1 - \sum_{|s|=n+1} L(s)$, and by definition $L(s) \geq 0$, it follows that $S(n) \leq 1$ as desired.

2. We have shown above that $\sum_{|s|\leq n} \Delta(L)(s) = S(n) = 1 - \sum_{|s|=n+1} L(s)$. So in the limit when n approaches inifinity, this gives us $\sum_s \Delta(L)(s) = 1 - \lim_{k\to\infty} \sum_{|s|=k} L(s)$, from which the result follows. ■

We conclude this section by defining a natural partial order on the set of p-languages under which it is a cpo. Continuity properties of various regular operators defined on the set of p-languages is then investigated with respect to this order in the next section. These continuity properties are then used to establish that recursive equations involving the operators are well defined.

Definition 3 Given a pair of p-languages $K, L \in \mathcal{L}$, define $K \preceq L$ if and only if

$$\forall s \in \Sigma^* : K(s) \leq L(s).$$

It is easy to see that the relation $\preceq$ as defined above is reflexive, anti-symmetric and transitive, i.e., it is a partial order. The infimum and supremum of $K, L \in \mathcal{L}$ are defined as follows:

$$K \sqcap L(s) = \inf(K(s), L(s)); \quad K \sqcup L(s) := \sup(K(s), L(s)).$$

Example 2 Consider p-languages H, K, L defined over $\Sigma = \{a, b\}$:

$H(\epsilon) = 1, H(a) = 0.4, H(ab) = 0.3, H(s) = 0,$ otherwise
$K(\epsilon) = 1, K(a) = 0.4, K(b) = 0.5, K(ab) = 0.4, K(s) = 0,$ otherwise
$L(\epsilon) = 1, L(a) = 0.6, L(b) = 0.4, L(ab) = 0.3, L(s) = 0,$ otherwise

Then $H \preceq K$, and $H \preceq L$, while K and L are incomparable. Note that $K \sqcap L$ is a p-language. However, $K \sqcup L$ is not a p-language, since $K \sqcup L(a) = 0.6$ and $K \sqcup L(b) = 0.5$ which violates P2.

It follows from the above example that the set of p-languages under partial order $\preceq$ is not a lattice. The following theorem shows that this set is an inf semi-lattice, and also a cpo.

Theorem 2 Consider the partial order $\preceq$ on $\mathcal{L}$ as defined above. Then

1. $(\mathcal{L}, \preceq)$ is an inf semi-lattice.
2. $(\mathcal{L}, \preceq)$ is a cpo.

Proof: 1. Consider $K, L \in \mathcal{L}$. We need to show that $K \sqcap L \in \mathcal{L}$, i.e., P1 and P2 hold for $K \sqcap L$. Clearly, $K \sqcap L(\epsilon) = \inf(K(\epsilon), L(\epsilon)) = 1$, which implies that P1 holds. In order to see that P2 also holds consider the following:

$$\begin{aligned}
\sum_\sigma K \sqcap L(s\sigma) &= \sum_\sigma \inf(K(s\sigma), L(s\sigma)) \\
& \quad \{\text{from property of addition and infimum}\} \\
&\leq \inf(\sum_\sigma K(s\sigma), \sum_\sigma L(s\sigma)) \\
& \quad \{\text{from P2}\} \\
&\leq \inf(K(s), L(s)) = K \sqcap L(s).
\end{aligned}$$

2. The bottom element for $\mathcal{L}$ is the nil language $I \in \mathcal{L}$. So next consider a chain of p-languages $\{L_i\}_{i \geq 0}$, i.e., $L_i \preceq L_j$ whenever $i \leq j$. Then for each $s \in \Sigma^*$, $\{L_i(s)\}$ is a monotonically increasing sequence of reals bounded above by 1, and hence converges. We use L_∞ to denote the resulting "trace-wise" limit of the chain of p-languages. We need to show that L_∞ itself is a p-language.

Clearly, $L_\infty(\epsilon) = \lim_{i \to \infty} L_i(\epsilon) = 1$, which implies that P1 holds. In order to see that P2 also holds consider the following:

$$\begin{aligned}
\sum_\sigma L_\infty(s\sigma) &= \sum_\sigma \lim_{i \to \infty} L_i(s\sigma) \\
& \quad \{\text{limit commutes with sum over finite terms}\} \\
&= \lim_{i \to \infty} \sum_\sigma L_i(s\sigma) \\
& \quad \{\text{from P2}\} \\
&\leq \lim_{i \to \infty} L_i(s) = L_\infty(s).
\end{aligned}$$

■

4 Operators for Probabilistic Languages

In this section we define some operations on p-languages and study their properties. These operators are useful in describing a complex system as a composition of many simple systems.

4.1 Choice

This operator captures non-deterministic choice between two systems. The composite system behaves as one of the two systems with certain given probabilities.

Definition 4 Given a pair of p-languages $L_1, L_2 \in \mathcal{L}$, and $e \in [0,1]$, the *choice* operation, denoted $L_1 +_e L_2$, is defined as:

$$L_1 +_e L_2 := eL_1 + \bar{e}L_2,$$

where $\bar{e} := 1 - e$.

In other words, the combined system either behaves as L_1 with probability e or as L_2 with probability $1 - e$. As a part of the next theorem we prove that $L_1 +_e L_2$ is a p-language. Note that the choice operator can be easily generalized for multiple p-languages, in which case it will be a convex combination of its argument p-languages.

In the following theorem we study a few properties of the choice operator.

Theorem 3 Consider $L_1, L_2 \in \mathcal{L}$ and $e \in [0,1]$.

1. $L_1 +_e L_2$ is a p-language.
2. $\Delta(L_1 +_e L_2) = e\Delta(L_1) + \bar{e}(\Delta(L_2)$.
3. Choice is a continuous operator in both of its arguments.

Proof: 1. That $L_1 +_e L_2$ is a p-language is clear since it is the convex combination of two p-languages.

2. This follows from the following series of equalities:

$$\begin{aligned}
\Delta(L_1 +_e L_2) &= L_1 +_e L_2 - \Lambda(L_1 +_e L_2) \\
&= (eL_1 + \bar{e}L_2) - \Lambda(eL_1 + \bar{e}L_2) \\
&\quad \{\text{from linearity of } \Lambda\} \\
&= (eL_1 + \bar{e}L_2) - [e\Lambda(L_1) + \bar{e}\Lambda(L_2)] \\
&= [eL_1 - e\Lambda(L_1)] + [\bar{e}L_2 - \bar{e}\Lambda(L_2)] \\
&= e\Delta(L_1) + \bar{e}\Delta(L_2).
\end{aligned}$$

3. By symmetry of the definition of the choice operator in its arguments, it suffices to show that for any chain $\{L_i\}$ of p-languages and $e \in [0,1]$:

$$K +_e (\sqcup_i L_i) = \sqcup_i (K +_e L_i).$$

Note that $\sqcup_i L_i$ is well defined from Theorem 2. Then the desired equality follows from the observation that for any $s \in \Sigma^*$:

$$eK(s) + \bar{e}(\lim_{i \to \infty} L_i)(s) = \lim_{i \to \infty}[eK(s) + \bar{e}L_i(s)],$$

where we have used the fact that the limit operation commutes with summation over a finite number of terms. ■

Since choice is continuous in both its arguments, it is possible to define a "choice function" mapping $\mathcal{L}$ to itself and obtain its infimum fixed point. Given $L \in \mathcal{L}$ and $e \in [0,1]$, we define the function $+_e L : \mathcal{L} \to \mathcal{L}$ as follows:

$$\forall H \in \mathcal{L} : +_L(H) := L +_e H.$$

Next we study the fixed point equation involving the choice function.

Theorem 4 Given $L \in \mathcal{L}$ and $e \in [0,1]$, the infimum fixed point of the equation $+_e L(H) = H$, i.e., $H = L +_e H$, where $H \in \mathcal{L}$ is the variable of the equation, is given by L when $e \neq 0$, and I when $e = 0$.

Proof: Since $(\mathcal{L}, \preceq)$ is a cpo, and the choice function $+_e L$ is continuous in this cpo, its infimum fixed point exists. Note that when $e = 0$, then for any $H \in \mathcal{L}$, $+_e L(H) = eL + \bar{e}H = H$, i.e., each p-language is a fixed point. So the infimum fixed point equals I. For the case when $e \neq 0$, we proceed as follows.

Note $L +_e L = eL + \bar{e}L = L$, implying that L is a fixed point. We need to show that it is the infimum fixed point. From a well know result on fixed points [3, 10], we know that the infimum fixed point can be computed as $\sqcup_{i \geq 0}(+_e L)^i(I)$, where the nil p-language I is the bottom element of $\mathcal{L}$. Define $H_n := \sqcup_{i \leq n}(+_e L)^i(I)$. Then it is easy to show using induction on n that

$$H_n = \sum_{i=0}^{n-1} \bar{e}^i eL + \bar{e}^n I.$$

Since $e \neq 0, \bar{e} \neq 1$, which implies $\bar{e}^n$ approaches zero as n approaches infinity. Hence the limiting value of H_n is given by

$$\sum_{i \geq 0} \bar{e}^i eL = \frac{1}{1 - \bar{e}} eL = \frac{1}{e} eL = L,$$

as desired. ■

4.2 Concatenation

This operator captures the sequential operation of two systems. The composite system behaves as one system, and upon termination continues to behave as the other system with a certain given probability.

Example 3 Suppose the p-languages L_1 and L_2 describe the processes of tossing a coin until a head and a tail, respectively, appear. Suppose the probability of getting a head in the first (resp., second) process is p_1 (resp., p_2). Then the event set for both the systems is given by $\Sigma = \{\text{toss}\}$, and

$$L_1(\text{toss}^n) = (1-p_1)^n; \quad \Delta(L_1)(\text{toss}^n) = (1-p_1)^n p_1$$

$$L_2(\text{toss}^n) = p_2^n; \quad \Delta(L_2)(\text{toss}^n) = p_2^n(1-p_2).$$

Next suppose the first coin is tossed until a head appears, and then with probability e the second coin is selected and tossed until the tail appears. Then the behavior of the composite process is described by the concatenation of the two processes, which we denote by $L_1 \cdot_e L_2$. It is clear that a sequence of n tosses will take place in the composite process if either (i) the first coin is tossed n times, always getting a tail, or (ii) the first coin is tossed $m < n$ times, always getting a tail, at which point the first process terminates (a head occurs next), and then with probability e the second coin is selected and tossed $n-m$ times, always getting a head. This motivates the following definition of concatenation.

Definition 5 Given a pair of p-languages $L_1, L_2 \in \mathcal{L}$, and $e \in [0,1]$, the *concatenation* operation, denoted $L_1 \cdot_e L_2$, is defined as:

$$\begin{aligned} \forall s \in \Sigma^* : L_1 \cdot_e L_2(s) \;&:=\; L_1(s) + e \sum_{t<s} \Delta(L_1)(t) L_2(t^{-1}s) \\ &=\; L_1(s) - e\Delta(L_1)(s) + e \sum_{t} \Delta(L_1)(t) L_2(t^{-1}s). \end{aligned}$$

In other words, the composite system executes a trace s by first behaving as the system with p-language L_1, and either executes the entire trace s that way, or terminates after executing a prefix t of s and then with probability e executes the remainder of the trace $t^{-1}s$ as the system with p-language L_2.

We show below that $L_1 \cdot_e L_2$ is indeed a p-language, i.e., it satisfies P1 and P2. We first prove a few properties of the "convolution" operation that appears in form of the term $\sum_t \Delta(L_1)(t) L_2(t^{-1}s)$ in the definition of concatenation. The following definition of convolution is introduced to simplify future notation.

Definition 6 Given a pair of real valued $f, g : \Sigma^* \to \mathcal{R}$, their *convolution*, denoted $f \circ g$, is defined as:

$$f \circ g(s) := \sum_t f(t) g(t^{-1}s).$$

Then the concatenation operation can be simply defined as:

$$L_1 \cdot_e L_2 = L_1 - e\Delta(L_1) + e\Delta(L_1) \circ L_2.$$

The convolution operator defined above is termed as product (or Cauchy product) in the theory of formal power series [20]. We use *convolution* to avoid any confusion with concatenation product.

The following lemma describes certain properties of the convolution operator.

Lemma 1 Let $f, g, h : \Sigma^* \to \mathcal{R}$ be real valued functions, and $e, e' \in \mathcal{R}$.

1. [1] I is the identity for convolution, i.e., $f \circ I = I \circ f = f$.
2. [1] Convolution is associative, i.e., $f \circ (g \circ h) = (f \circ g) \circ h$.
3. Convolution is continuous in both arguments, i.e., $f \circ (\sqcup_i g_i) = \sqcup_i (f \circ g_i)$, and $(\sqcup_i f_i) \circ g = \sqcup_i (f_i \circ g)$.
4. [1] Convolution is a linear operator, i.e., $f \circ (eg + e'h) = ef \circ g + e'f \circ h$.
5. If $g(\epsilon) = 1$, then $\Lambda(f \circ g) = f \circ \Lambda(g) + \Lambda(f)$.

Proof: The proofs for parts 1, 2, and 4 can be found in [1]. So we only prove parts 3 and 5.

3. The following holds for each $s \in \Sigma^*$:

$$\begin{aligned}
f \circ (\sqcup_i g_i(s)) &= \sum_t f(t)(\lim_{i\to\infty} g_i(t^{-1}s)) \\
&\quad \{\text{limit commutes with sum over finite terms (all prefixes of } s)\} \\
&= \lim_{i\to\infty} \sum_t f(t) g_i(t^{-1}s) \\
&= \sqcup_i (f \circ g_i(s)),
\end{aligned}$$

The continuity in the other argument can be similarly proved.

5. The following holds for any $s \in \Sigma^*$:

$$\begin{aligned}
\Lambda(f \circ g)(s) &= \sum_\sigma f \circ g(s\sigma) \\
&= \sum_\sigma \sum_{t \le s\sigma} f(t) g(t^{-1}s\sigma) \\
&= \sum_\sigma \left[\sum_{t \le s} f(t) g(t^{-1}s\sigma) + f(s\sigma) g(\epsilon) \right] \\
&\quad \{\text{since } g((s\sigma)^{-1}s\sigma) = g(\epsilon) = 1\} \\
&= \sum_{t \le s} f(t) \sum_\sigma g(t^{-1}s\sigma) + \sum_\sigma f(s\sigma) \\
&= \sum_{t \le s} f(t) \Lambda(g)(t^{-1}s) + \Lambda(f)(s) \\
&= f \circ \Lambda(g)(s) + \Lambda(f)(s).
\end{aligned}$$

$\blacksquare$

In the following theorem we study a few properties of the concatenation operator.

Theorem 5 Consider $L_1, L_2 \in \mathcal{L}$ and $e \in [0,1]$.

1. $L_1 \cdot_e L_2$ is a p-language.
2. $\Delta(L_1 \cdot_e L_2) = \bar{e}\Delta(L_1) + e\Delta(L_1) \circ \Delta(L_2)$.
3. Concatenation is continuous in its second argument; it is not even monotone in its first argument.

Proof: 1. P1 follows from the following:

$$L_1 \cdot_e L_2(\epsilon) = L_1(\epsilon) - e\Delta(L_1)(\epsilon) + e\Delta(L_1) \circ L_2(\epsilon) = L_1(\epsilon) - e\Delta(L_1)(\epsilon) + e\Delta(L_1)(\epsilon)L_2(\epsilon) = 1.$$

In order to prove P2 it suffices to show that $\Delta(L_1 \cdot_e L_2) \geq 0$. This follows from the second part below.

2. We have the following series of equalities:

$$\begin{aligned}
\Delta(L_1 \cdot_e L_2) &= L_1 \cdot_e L_2 - \Lambda(L_1 \cdot_e L_2) \\
&= [L_1 - e\Delta(L_1) + e\Delta(L_1) \circ L_2] - \Lambda[L_1 - e\Delta(L_1) + e\Delta(L_1) \circ L_2] \\
&\quad \{\text{using linearity of } \Lambda \text{ and rearranging}\} \\
&= [L_1 - \Lambda(L_1)] + e[\Delta(L_1) \circ L_2 - \Delta(L_1) + \Lambda(\Delta(L_1)) - \Lambda(\Delta(L_1) \circ L_2)] \\
&\quad \{\text{from Lemma 1 } \Lambda(\Delta(L_1) \circ L_2) = \Delta(L_1) \circ \Lambda(L_2) + \Lambda(\Delta(L_1))\} \\
&= \Delta(L_1) + e[\Delta(L_1) \circ L_2 - \Delta(L_1) + \Lambda(\Delta(L_1)) - \Delta(L_1) \circ \Lambda(L_2) - \Lambda(\Delta(L_1))] \\
&= \Delta(L_1) + e[\Delta(L_1) \circ L_2 - \Delta(L_1) - \Delta(L_1) \circ \Lambda(L_2)] \\
&\quad \{\text{from linearity of convolution}\} \\
&= [\Delta(L_1) - e\Delta(L_1)] + e\Delta(L_1) \circ [L_2 - \Lambda(L_2)] \\
&= \bar{e}\Delta(L_1) + e\Delta(L_1) \circ \Delta(L_2).
\end{aligned}$$

3. Consider $K \in \mathcal{L}$ and a chain of p-languages $\{L_i\}$. Then we have the following series of equalities:

$$\begin{aligned}
K \cdot_e \sqcup L_i &= K - e\Delta(K) + e\Delta(K) \circ \sqcup_i L_i \\
&\quad \{\text{from continuity of convolution, Lemma 1, part 3}\} \\
&= K - e\Delta(K) + e \sqcup_i \Delta(K) \circ L_i \\
&\quad \{\text{limit commutes with sum over finite terms}\} \\
&= \sqcup_i K - e\Delta(K) + e\Delta(K) \circ L_i \\
&= \sqcup_i K \cdot_e L_i,
\end{aligned}$$

In order to see that concatenation is not even monotone in its first argument, consider p-languages K, L over the event set $\Sigma = \{a, b\}$ with

$K(\epsilon) = 1, K(a) = 0.2, K(s) = 0,$ otherwise
$L(\epsilon) = 1, L(b) = 1.0, L(s) = 0,$ otherwise.

Then $I \preceq K$, but $I._1L$ and $K._1L$ are incomparable since $I._1L = L$ and $K._1L$ is given by:

$$K._1L(\epsilon) = 1, K._1L(a) = .2, K._1L(b) = .8, K._1L(ab) = .2, K._1L(s) = 0, \text{ otherwise.}$$

This completes the proof. ■

Remark 4 In order to understand why concatenation is not continuous in its first argument consider a chain of p-languages $\{L_i\}$, a p-language $K \in \mathcal{L}$, and $e \in [0,1]$. Then

$$\sqcup_i L_i ._e K = \sqcup_i L_i - e\Delta(\sqcup_i L_i) + e\Delta(\sqcup_i L_i) \circ K.$$

Thus for the concatenation to be continuous in its first argument, we need $\Delta(\sqcup_i L_i) = \sqcup_i \Delta(L_i)$, i.e., we need the limiting completion probability function to be the limit of the individual completion probability functions. However, although the given p-language form a chain, their completion probability functions may not, and so their limit may not even exist. This causes the loss of continuity of concatenation in its first argument.

Since concatenation is continuous in its second argument, it is possible to define a "concatenation function" mapping $\mathcal{L}$ to itself and obtain its infimum fixed point. Given $L \in \mathcal{L}$ and $e \in [0,1]$, we define the function $._eL : \mathcal{L} \to \mathcal{L}$ as follows:

$$\forall H \in \mathcal{L} : ._eL(H) := L._eH.$$

Next we study the fixed point equation involving the concatenation function.

Theorem 6 Given $L \in \mathcal{L}$ and $e \in [0,1]$, the infimum fixed point of the equation $._eL(H) = H$, i.e., $H = L._eH$, is given by $\sum_{i\geq 0} e^i \Delta(L)^{(i)} \circ L - \sum_{i\geq 1} e^i \Delta(L)^{(i)}$, where $\Delta(L)^{(i)}$ represents the i-fold convolution of $\Delta(L)$ with itself, and $\Delta(L)^{(0)} := I$.

Proof: Since $(\mathcal{L}, \preceq)$ is a cpo, and the concatenation function $._eL$ is continuous on this cpo, its infimum fixed point exists. From a well know fixed point result [3, 10], the infimum fixed point is given by $\sqcup_{i\geq 0}(._eL)^i(I)$, where I is the nil p-language which is also the bottom element of $\mathcal{L}$.

Define $H_n := \sqcup_{i\leq n}(._eL)^i(I)$. We show using induction on n that

$$H_n = \sum_{i=0}^{n-1} e^i \Delta(L)^{(i)} \circ L - \sum_{i=1}^{n-1} e^i \Delta(L)^{(i)}, \tag{2}$$

from which the result follows by taking the limit as n approaches infinity. For the base step consider $n = 1$. Then by definition

$$H_1 = L._eI = L - e\Delta(L) + e\Delta(L) \circ I = L - e\Delta(L) + e\Delta(L) = L,$$

where the last equality follows from the fact that I is the identity of convolution (Lemma 1). Also, when $n = 1$, (2) reduces to:

$$H_1 = e^0\Delta(L)^{(0)} \circ L = I \circ L = L,$$

which establishes the base step. In order to see the induction step we have:

$$\begin{aligned}
H_{n+1} &= L._e H_n \\
&= L - e\Delta(L) + e\Delta(L) \circ H_n \\
&\quad \{\text{from induction hypothesis}\} \\
&= L - e\Delta(L) + e\Delta(L) \circ [\sum_{i=0}^{n-1} e^i\Delta(L)^{(i)} \circ L - \sum_{i=1}^{n-1} e^i\Delta(L)^{(i)}] \\
&= L - e\Delta(L) + \sum_{i=1}^{n} e^i\Delta(L)^{(i)} \circ L - \sum_{i=2}^{n} e^i\Delta(L)^{(i)} \\
&= [L + \sum_{i=1}^{n} e^i\Delta(L)^{(i)} \circ L] - [e\Delta(L) + \sum_{i=2}^{n} e^i\Delta(L)^{(i)}] \\
&= \sum_{i=0}^{n} e^i\Delta(L)^{(i)} \circ L - \sum_{i=1}^{n} e^i\Delta(L)^{(i)}.
\end{aligned}$$

This establishes the induction step and completes the proof. ∎

Remark 5 Since the choice and concatenation functions as defined above are both continuous, any function formed using them is also continuous and thus possesses the infimum fixed point. As an example consider the function $f : \mathcal{L} \to \mathcal{L}$ defined as $f(H) = L +_1 (K._1 H)$ for each $H \in \mathcal{L}$, where $K, L \in \mathcal{L}$ are fixed p-languages. Then its infimum fixed point can be computed using the technique illustrated above.

4.3 Concatenation Closure

The infimum fixed point of the concatenation function allows us to define the concatenation closure operator. This describes a system which upon termination continues with a certain probability e to behave as itself any finite number of times. This operation resembles the Kleene closure operation from the theory of formal languages and captures the notion of repeated sequential behavior.

Definition 7 Given a p-language $L \in \mathcal{L}$ and $e \in [0, 1]$, the *concatenation closure* operation, denoted L^{*_e}, is defined as:

$$L^{*_e} := \sum_{i \geq 0} e^i\Delta(L)^{(i)} \circ L - \sum_{i \geq 1} e^i\Delta(L)^{(i)} = L + \sum_{i \geq 1} e^i\Delta(L)^{(i)} \circ [L - I],$$

i.e., it is the infimum fixed point of the concatenation function $._e L$

Since L^{*_e} as defined above is the infimum fixed point of the concatenation function (Theorem 6), it follows that it is a p-language, i.e., satisfies P1 and P2. The $(n+1)$th term of the summation in the definition of concatenation closure is $e^n\Delta(L)^{(n)} \circ [L-I]$, which represents that the system repeatedly behaves as itself n different times—terminating each time (represented by the n-fold convolution term $e^n\Delta(L)^{(n)}$)—and then continues as itself for the $n+1$th time without terminating and executing a "non-epsilon trace" (represented by the final convolution term $L-I$).

In the following theorem we obtain a property of the concatenation closure operator.

Theorem 7 Consider a p-language $L \in \mathcal{L}$ and $e \in [0,1]$. Then $\Delta(L^{*_e}) = \frac{\bar{e}}{e}\sum_{i\geq 1} e^i\Delta(L)^{(i)}$.

Proof: For notational simplicity define $(e\Delta(L))^{(+)} := \sum_{i\geq 1} e^i\Delta(L)^{(i)}$. Then $L^{*_e} = L + (e\Delta(L))^{(+)} \circ (L-I)$. So

$$
\begin{aligned}
& \Delta(L*_e) \\
= \; & L + (e\Delta(L))^{(+)} \circ (L-I) - \Lambda(L + (e\Delta(L))^{(+)} \circ (L-I)) \\
& \{\text{from linearity of } \Lambda\} \\
= \; & L + (e\Delta(L))^{(+)} \circ (L-I) - \Lambda(L) - \Lambda((e\Delta(L))^{(+)} \circ L) + \Lambda((e\Delta(L))^{(+)}) \\
& \{\text{from Lemma 1, parts 1 and 5}\} \\
= \; & L + [(e\Delta(L))^{(+)} \circ L - (e\Delta(L))^{(+)}] - \Lambda(L) - [(e\Delta(L))^{(+)} \circ \Lambda(L) + \Lambda((e\Delta(L))^{(+)})] \\
& +\Lambda((e\Delta(L))^{(+)}) \\
= \; & [L - \Lambda(L)] + [(e\Delta(L))^{(+)} \circ (L - \Lambda(L))] - (e\Delta(L))^{(+)} \\
= \; & \Delta(L) + (e\Delta(L))^{(+)} \circ \Delta(L) - (e\Delta(L))^{(+)} \\
= \; & \frac{1}{e}[e\Delta(L) + e\sum_{i\geq 1} e^i\Delta(L)^{(i)} \circ \Delta(L)] - (e\Delta(L))^{(+)} \\
= \; & \frac{1}{e}[e\Delta(L) + \sum_{i\geq 2} e^i\Delta(L)^{(i)}] - (e\Delta(L))^{(+)} \\
= \; & \frac{1}{e}[\sum_{i\geq 1} e^i\Delta(L)^{(i)}] - (e\Delta(L))^{(+)} \\
= \; & [\frac{1}{e} - 1](e\Delta(L))^{(+)} \\
= \; & \frac{\bar{e}}{e}(e\Delta(L))^{(+)},
\end{aligned}
$$

as desired. ■

Example 4 Consider a process of tossing a coin which is repeated till a head occurs. Suppose that the probability of getting a tail is e. Then in this example, the "toss once" process is repeated each time a tail occurs, i.e., with probability e. So the "repeated toss" process can be represented as the concatenation-closure of the "toss once" process. For the "toss once" process $\Sigma = \{\text{toss}\}$, and

$L(\epsilon) = L(\text{toss}) = 1, L(s) = 0, \text{ otherwise.}$

The "repeated toss" process can be described by the p-language L^{*_e}.

Since $\Delta(L)(s) = 1$ if and only if $s = \text{toss}$, and zero otherwise, it follows easily from induction that $\Delta(L)^{(n)}(s) = 1$ if and only if $s = \text{toss}^n$. Suppose we are interested in finding the probability of n tosses. Then this is given by:

$$\begin{aligned} L^{*_e}(\text{toss}^n) &= \left[\sum_{i\geq 0} e^i \Delta(L)^{(i)} \circ L\right](\text{toss}^n) - \left[\sum_{i\geq 1} e^i \Delta(L)^{(i)}\right](\text{toss}^n) \\ &= [e^{n-1}\Delta(L)^{(n-1)}(\text{toss}^{n-1})L(\text{toss}) + e^n\Delta(L)^{(n)}(\text{toss}^n)L(\epsilon)] - [e^n\Delta(L)^{(n)}(\text{toss}^n)] \\ &= e^{n-1} + e^n - e^n \\ &= e^{n-1}, \end{aligned}$$

which is precisely the probability of getting $n-1$ consecutive tails.

Similarly, if we are interested in determining the probability of termination after n tosses, then this is given by:

$$\begin{aligned} \Delta(L^{*_e})(\text{toss}^n) &= \frac{\bar{e}}{e}\sum_{i\geq 1} e^i \Delta(L)^{(i)}(\text{toss}^n) \\ &= \frac{\bar{e}}{e} e^n \Delta(L)^n(\text{toss}^n) \\ &= \frac{\bar{e}}{e} e^n \\ &= e^{n-1}\bar{e}, \end{aligned}$$

which is precisely the probability of getting $n-1$ consecutive tails followed by a head.

5 Automata for Probabilistic Languages

In this section we define "probabilistic automata"—nondeterministic automata with probabilities associated with its transitions, which can be used to represent p-languages in a concise manner. It is easier represent a stochastic discrete event system as a probabilistic automaton.

Definition 8 A *probabilistic automaton* (p-automaton) G over the event set Σ is a triple $G := (X, x_0, P)$, where X is the set of states, $x_0 \in X$ is the initial state, and $P : X \times \Sigma \times X \to [0, 1]$ is the transition probability function satisfying

C1: $\forall x \in X : \sum_{y\in X} \sum_{\sigma\in\Sigma} P(x, \sigma, y) \leq 1$.

G is said to be deterministic if for each $x \in X$ and $\sigma \in \Sigma$, there exists at most one $y \in X$ such that $P(x, \sigma, y) > 0$.

Such a p-automata starts from its initial state x_0, and when at state x it moves to state y on event σ with probability $P(x,\sigma,y)$. For each $x \in X$, we define $\Delta(G)(x) := 1 - \sum_{\sigma\in\Sigma}\sum_{y\in X} P(x,\sigma,y)$ to be the *probability of termination at state x*.

Next we define the p-language "generated by" such a p-automaton G. The transition probability function can be extended to paths in G, where a *path* in G is defined to be a member of $X(\Sigma X)^*$. In other words, a path is obtained by concatenating transitions where the end and start states of the consecutive transitions are the same. Given a path $\pi = x_0\sigma_1 x_1 \ldots \sigma_n x_n$, we use $|\pi| = n$ to denote its length; for each $k \leq |\pi|$, $\pi^k := x_0\sigma_1 x_1 \ldots \sigma_k x_k$ to be the initial sub-path of length k; and $tr(\pi) := \sigma_1 \ldots \sigma_n$ to be the event trace associated with path π. The probability measure for paths is inductively defined as:

$$\forall x \in X : P(x) = 1;$$
$$\forall \pi \in X(\Sigma X)^*, \sigma \in \Sigma, y \in X : P(\pi\sigma y) := P(\pi)P(x_{|\pi|},\sigma,y)$$

In other words, the probability measure of a path is the product of probability measures of the individual transitions constituting the path.

With these preliminaries we next define the p-language generated by a p-automaton.

Definition 9 Given a p-automaton $G := (X, x_0, P)$, the *p-language generated by* G, denoted L_G, is defined as:

$$L_G(s) := \sum_{\pi:tr(\pi)=s,\pi^0=x_0} P(\pi).$$

The following proposition states that L is indeed a p-language.

Theorem 8 Let $G := (X, x_0, P)$ be a p-automaton over Σ. Then L_G is a p-language.

Proof: From definition, we have $L_G(\epsilon) = P(x_0) = 1$, which demonstrates P1. In order to see P2, consider a trace $s \in \Sigma^*$ of length, say, n. Then

$$\begin{aligned}
\sum_\sigma L_G(s\sigma) &= \sum_\sigma \sum_{\pi:tr(\pi)=s\sigma,\pi^0=x_0} P(\pi) \\
&= \sum_\sigma \sum_{\bar\pi:tr(\bar\pi)=s,\bar\pi^0=x_0} \sum_{y\in X} P(\bar\pi)P(x_n(\bar\pi),\sigma,y) \\
&= \sum_{\bar\pi:tr(\bar\pi)=s,\bar\pi^0=x_0} P(\bar\pi) \sum_{y\in X}\sum_\sigma P(x_n(\bar\pi),\sigma,y) \\
& \quad \{\text{from C1}\} \\
&\leq \sum_{\bar\pi:tr(\bar\pi)=s,\bar\pi^0=x_0} P(\bar\pi) \\
&= L_G(S).
\end{aligned}$$

■

Remark 6 It follows that every p-automaton defines a p-language. Conversely, every p-language can be represented by a p-automaton: Given a p-language L, define a p-automaton $G := (\Sigma^*, \epsilon, P)$, where

$$\forall s, t \in \Sigma^*, \sigma \in \Sigma : P(s, \sigma, t) := \begin{cases} \frac{L(t)}{L(s)} & \text{if } t = s\sigma \\ 0 & \text{otherwise} \end{cases}$$

Since $L(s) \geq L(t)$ whenever $s \leq t$ (Remark 3), in the definition of transition probability $P(s, \sigma, t) \leq 1$. Then it is easy to verify that $L_G = L$ and that C1 holds for G. Also note that G is a deterministic p-automaton.

A special class of p-languages which is of interest is the class that can be represented by a p-automaton with finitely many states, which we call the class of regular p-languages.

Definition 10 A p-language $L \in \mathcal{L}$ is said to be *regular* if there exists a p-automaton G with finitely many states such that $L_G = L$.

The next theorem shows that the operators choice, concatenation, and concatenation-closure preserve regularity.

Theorem 9 Consider regular p-languages L_1, L_2 and $e \in [0, 1]$. Then $L_1 +_e L_2$, $L_1 ._e L_2$, and $L_1^{*_e}$ are also regular p-languages.

Proof: For $i = 1, 2$, let $G_i := (X_i, x_{0,i}, P_i)$ be finite p-automata with $L_{G_i} = L_i$.

In order to show the regularity of $L_1 +_e L_2$, define a finite p-automaton $G := (X, x_0, P)$, where the state set is $X := X_1 \cup X_2 \cup \{x_0\}$, x_0 is a new state which is the initial state of G, and the state transition probabilities are given by:

$$\forall x, y \in X, \sigma \in \Sigma : P(x, \sigma, y) := \begin{cases} P_1(x, \sigma, y) & \text{if } x, y \in X_1 \\ P_2(x, \sigma, y) & \text{if } x, y \in X_2 \\ eP_1(x_{0,1}, \sigma, y) & \text{if } x = x_0, y \in X_1 \\ \bar{e}P_2(x_{0,2}, \sigma, y) & \text{if } x = x_0, y \in X_2 \\ 0 & \text{otherwise} \end{cases}$$

Then it is easy to see that G is a p-automaton, and $L_G = L_1 +_e L_2$.

Next to show the regularity of $L_1 ._e L_2$, define a finite p-automaton $G := (X, x_{0,1}, P)$, where the state set $X := X_1 \cup X_2$, and the state transition probabilities are given by:

$$\forall x, y \in X, \sigma \in \Sigma : P(x, \sigma, y) := \begin{cases} P_1(x, \sigma, y) & \text{if } x, y \in X_1 \\ P_2(x, \sigma, y) & \text{if } x, y \in X_2 \\ e[\Delta(G_1)(x)]P_2(x_{0,2}, \sigma, y) & \text{if } x \in X_1, y \in X_2 \\ 0\text{otherwise} \end{cases}$$

Then it is easy to see that G is a p-automaton, and $L_G = L_1 ._e L_2$.

Finally to show the regularity of $L_1^{*_e}$ define a finite p-automaton $G := (X_1, x_{0,1}, P)$, where the state transition probabilities are given by:

$$\forall x, y \in X_1, \sigma \in \Sigma : P(x, \sigma, y) := P_1(x, \sigma, y) + e[\Delta(G_1)(x)]P_1(x_{0,1}, \sigma, y).$$

Then it is easy to see that G is a p-automaton, and $L_G = L_1^{*_e}$. ■

6 Performance Functions

In order to study the performance of a stochastic discrete event system, we use real valued functions defined over the set of event traces to associate a cost with each trace of the system, and we compute the average value of the cost with respect to the completion probability function. Recall that the completion probability function associates a probability of termination with each trace of the system. Since the outcomes corresponding to terminations following distinct traces are all mutually exclusive, the cumulative probability of termination is obtained by simply adding the completion probabilities of the individual traces. This cumulative probability equals one if and only if the system is terminating (Theorem 1), in which case the completion probability function defines a *probability mass function* [5] on the set of traces. So we assume in this section that the systems under investigation are terminating, and use it to give the definition of the average performance of a system.

Definition 11 Given a terminating system with p-language $L \in \mathcal{L}$, i.e., $\sum_s \Delta(L)(s) = 1$, and a performance function $F : \Sigma^* \to \mathcal{R}$ representing cost associated with each trace, the average value of F with respect to the completion probability mass function, denoted $E[F, \Delta(L)] \in \mathcal{R}$, is given by $E[F, \Delta(L)] := \sum_s \Delta(L)(s)F(s)$.

In the following subsections we consider two special types of performance functions and study how their values for complex systems can be efficiently obtained from those of component subsystems.

6.1 Additive Performance Function

Let $F : \Sigma^* \to \mathcal{R}$ be an "additive performance function" satisfying the property:

F1: $F(st) = F(s) + F(t)$

An example of an additive performance function is the completion time (or the first moment of completion time if it is a random variable [14]) which gives the time consumed in execution of a trace.

The following lemma lists a few properties of the average of an additive performance function.

Lemma 2 Consider an additive performance function $F : \Sigma^* \to \mathcal{R}$, and terminating p-languages $L_1, L_2, L \in \mathcal{L}$ having completion probability functions C_1, C_2, C, respectively. Then

1. $E[F, \Delta(L_1) \circ \Delta(L_2)] = E[F, \Delta(L_1)] + E[F, \Delta(L_2)]$
2. $E[F, \Delta(L)^{(n)}] = nE[F, \Delta(L)]$

Proof: We only prove the first part since the second follows from the first. We have the following series of equality:

$$
\begin{aligned}
E[F, \Delta(L_1) \circ \Delta(L_2)] &= \sum_s \Delta(L_1) \circ \Delta(L_2)(s) F(s) \\
& \{s = t.t^{-1}s \text{ when } t \leq s\} \\
&= \sum_s [\sum_t \Delta(L_1)(t) \Delta(L_2)(t^{-1}s)] F(t.t^{-1}s) \\
& \{\text{changing order of summation and using additivity of } F\} \\
&= \sum_t \sum_s \Delta(L_1)(t) \Delta(L_2)(t^{-1}s)[F(t) + F(t^{-1}s)] \\
& \{\text{applying change of variables } u := t^{-1}s\} \\
&= \sum_t \sum_u \Delta(L_1)(t) \Delta(L_2)(u)[F(t) + F(u)] \\
&= \sum_t \sum_u \Delta(L_1)(t) \Delta(L_2)(u) F(t) + \sum_t \sum_u \Delta(L_1)(t) \Delta(L_2)(u) F(u) \\
&= \sum_u \Delta(L_2)(u)[\sum_t \Delta(L_1)(t) F(t)] + \sum_t \Delta(L_1)(t)[\sum_u \Delta(L_2)(u) F(u)] \\
& \{\sum_u \Delta(L_2)(u) = \sum_t \Delta(L_1)(t) = 1\} \\
&= E[F, \Delta(L_1)] + E[F, \Delta(L_2)].
\end{aligned}
$$

■

The result of the above lemma is used in the next theorem to obtain average performance of a complex system in terms of those of the constituent subsystems.

Theorem 10 Consider an additive performance function $F : \Sigma^* \to \mathcal{R}$, and terminating p-languages $L_1, L_2, L \in \mathcal{L}$.

1. $E[F, \Delta(L_1 +_e L_2)] = eE[F, \Delta(L_1)] + \bar{e}E[F, \Delta(L_2)]$
2. $E[F, \Delta(L_1 ._e L_2)] = E[F, \Delta(L_1)] + eE[F, \Delta(L_2)]$
3. $E[F, \Delta(L^{*_e})] = \frac{1}{\bar{e}} E[F, \Delta(L)]$, whenever $e < 1$

Proof: 1. By Theorem 3, $\Delta(L_1 +_e L_2) = e\Delta(L_1) + \bar{e}\Delta(L_2)$. So

$$
\begin{aligned}
E[F, \Delta(L_1 +_e L_2)] &= \sum_s [e\Delta(L_1)(s)F(s) + \bar{e}\Delta(L_2)(s)F(s)] \\
&= e \sum_s \Delta(L_1)(s)F(s) + \bar{e} \sum_s \Delta(L_2)(s)F(s) \\
&= eE[F, \Delta(L_1)] + \bar{e}E[F, \Delta(L_2)].
\end{aligned}
$$

2. By Theorem 5, $\Delta(L_1 ._e L_2) = \bar{e}\Delta(L_1) + e\Delta(L_1) \circ \Delta(L_2)$. So from the first part above, $E[F, \Delta(L_1 ._e L_2)] = \bar{e}E[F, \Delta(L_1)] + eE[F, \Delta(L_1) \circ \Delta(L_2)]$. Since F is additive, from Lemma 2,

$E[F, \Delta(L_1) \circ \Delta(L_2)] = E[F, \Delta(L_1)] + E[F, \Delta(L_2)]$, from which the result follows.

3. By Theorem 7, $\Delta(L^{*_e}) = \frac{\bar{e}}{e} \sum_{i \geq 1} e^i \Delta(L)^{(i)}$. So from the first part of Lemma 2,

$$\begin{aligned}
E[F, \Delta(L^{*_e})] &= \frac{\bar{e}}{e} \sum_{i \geq 1} e^i E[F, \Delta(L)^{(i)}] \\
&\quad \{\text{from Lemma 2, part 2}\} \\
&= \frac{\bar{e}}{e} \sum_{i \geq 1} e^i i E[F, \Delta(L)] \\
&= \frac{\bar{e}}{e} E[F, \Delta(L)] \sum_{i \geq 1} e^i i \\
&\quad \{\sum_{i \geq 1} e^i i = \frac{e}{\bar{e}^2} \text{ whenever } e < 1\} \\
&= \frac{1}{\bar{e}} E[F, \Delta(L)].
\end{aligned}$$

■

One can use the results derived above to study the average completion time of timed discrete event systems, where transition probabilities as well as transition occurrence times are known.

6.2 Multiplicative Performance Function

Let $F : \Sigma^* \to \mathcal{R}$ be a multiplicative performance function satisfying the property:

F2: $F(st) = F(s)F(t)$

An example of a multiplicative performance function is the reliability function (or the first moment of the reliability function if it is a random variable) which gives the probability that a system does not fail, i.e., operates reliably, after executing a trace.

The following lemma lists a few properties of the average of a multiplicative performance function.

Lemma 3 Consider a multiplicative performance function $F : \Sigma^* \to \mathcal{R}$, and terminating p-languages $L_1, L_2, L \in \mathcal{L}$.

1. $E[F, \Delta(L_1) \circ \Delta(L_2)] = E[F, \Delta(L_1)]E[F, \Delta(L_2)]$
2. $E[F, \Delta(L)^{(n)}] = (E[F, \Delta(L)])^n$

Proof: Since the second part follows from the first part, we only prove the first part. We have the following series of equality:

$$E[F, \Delta(L_1) \circ \Delta(L_2)] = \sum_s \Delta(L_1) \circ \Delta(L_1)(s)F(s)$$

$$
\begin{aligned}
& \{s = t.t^{-1}s \text{ when } t \leq s\} \\
= \;& \sum_s [\sum_t \Delta(L_1)(t)\Delta(L_2)(t^{-1}s)] F(t.t^{-1}s) \\
& \{\text{changing order of summation and using multiplicativity of } F\} \\
= \;& \sum_t \sum_s \Delta(L_1)(t)\Delta(L_2)(t^{-1}s)[F(t)F(t^{-1}s)] \\
& \{\text{applying change of variables } u := t^{-1}s\} \\
= \;& \sum_t \sum_u \Delta(L_1)(t)\Delta(L_2)(u)[F(t)F(u)] \\
= \;& [\sum_t \Delta(L_1)(t)F(t)][\sum_u \Delta(L_2)(u)F(u)] \\
= \;& E[F, \Delta(L_1)]E[F, \Delta(L_2)].
\end{aligned}
$$

■

The result of the above lemma is used in the next theorem to obtain average performance of a complex system in terms of those of the constituent subsystems.

Theorem 11 Consider a multiplicative performance function $F : \Sigma^* \to \mathcal{R}$, and terminating p-languages $L_1, L_2, L \in \mathcal{L}$. Then

1. $E[F, \Delta(L_1 +_e L_2)] = eE[F, \Delta(L_1)] + \bar{e}E[F, \Delta(L_2)]$
2. $E[F, \Delta(L_1 ._e L_2)] = \bar{e}E[F, \Delta(L_1)] + eE[F, \Delta(L_1)]E[F, \Delta(L_2)]$
3. $E[F, \Delta(L^{*_e})] = \frac{\bar{e}E[F,\Delta(L)]}{1-eE[F,\Delta(L)]}$, whenever $eE[F, \Delta(L)] < 1$.

Proof: 1. The proof of this part is the same as that for the first part of Theorem 10.

2. By Theorem 5, $\Delta(L_1 ._e L_2) = \bar{e}\Delta(L_1) + e\Delta(L_1) \circ \Delta(L_2)$. So from the first part above, $E[F, \Delta(L_1 ._e L_2)] = \bar{e}E[F, \Delta(L_1)] + eE[F, \Delta(L_1) \circ \Delta(L_2)]$. Since F is multiplicative, from Lemma 3, $E[F, \Delta(L_1) \circ \Delta(L_2)] = E[F, \Delta(L_1)]E[F, \Delta(L_2)]$, from which the result follows.

3. By Theorem 7, $\Delta(L^{*_e}) = \frac{\bar{e}}{e} \sum_{i \geq 1} e^i \Delta(L)^{(i)}$. So from the first part of Lemma 3,

$$
\begin{aligned}
E[F, \Delta(L^{*_e})] \;=\; & \frac{\bar{e}}{e} \sum_{i \geq 1} e^i E[F, \Delta(L)^{(i)}] \\
& \{\text{from Lemma 3, part 2}\} \\
=\; & \frac{\bar{e}}{e} \sum_{i \geq 1} e^i (E[F, \Delta(L)])^i \\
& \{\sum_{i \geq 1} e^i (E[F, \Delta(L)])^i = \frac{eE[F, \Delta(L)]}{1 - eE[F, \Delta(L)]} \text{ whenever } eE[F, \Delta(L)] < 1\} \\
=\; & \frac{\bar{e}}{e} \left[\frac{eE[F, \Delta(L)]}{1 - eE[F, \Delta(L)]} \right] \\
=\; & \frac{\bar{e}E[F, \Delta(L)]}{1 - eE[F, \Delta(L)]}.
\end{aligned}
$$

■

Remark 7 The results of Theorems 10 and 11 can be combined to obtain compositional results for more general performance functions that are "bilinear". Suppose for example $F(st) = F(s) + F(t) + 2F(s)F(t)$, which is satisfied by the second moment of a random completion time [14]. Then using the techniques of Theorems 10 and 11 it can be shown that

1. $E[F, \Delta(L_1) \circ \Delta(L_2)] = E[F, \Delta(L_1)] + E[F, \Delta(L_2)] + 2E[F, \Delta(L_1)]E[F, \Delta(L_2)]$
2. $E[F, \Delta(L_1 +_e L_2)] = eE[F, \Delta(L_1)] + \bar{e}E[F, \Delta(L_2)]$
3. $E[F, \Delta(L_1 \cdot_e L_2)] = E[F, \Delta(L_1)] + eE[F, \Delta(L_2)]\{1 + 2E[F, \Delta(L_1)]\}$

7 Conclusion and Discussion

In this paper we have introduced a formalism for the specification and analysis of qualitative behavior of stochastic discrete event systems which are typically modeled as nondeterministic automata with probabilities associated with transitions. We have shown that systems defined in our formalism form a complete partial order under a suitable ordering relation. The regular operators choice, concatenation, and concatenation closure have been defined in this formalism which allow representation of complex systems as combination of simpler ones. These operators are continuous in the complete partial order of systems. Consequently it is easy to define systems using recursions involving these operators. It is also shown that the space of these systems is a complete metric space in which the operators are continuous and contracting. So the uniqueness of the fixed points of recursive equations follows from the contraction mapping theorem. Various average performance measures of stochastic discrete event systems can be defined in this formalism, and we provide analytical techniques to aide their computations.

Our formalism is also suitable for studying supervisory control problems of stochastic discrete event systems. We discuss a few ways in which control problems can be formulated in this setting. A detailed examination of these problems is subject of work in progress. Following the framework of supervisory control proposed by Ramadge and Wonham [18], the event set is partitioned into the set of controllable and uncontrollable events. The supervisor is not allowed to disable the uncontrollable events, but can disable any of the controllable events. One possibility is that the probability of the disabled events is added to the termination event probability. In other words, the supervisor terminates the system if any of the disabled event is chosen. Another possibility is that the probabilities of the enabled events (including that of the termination event) is renormalized using the probabilities of the disabled events. In other words, the probability of an enabled event is the conditional probability of its occurrence subject to the condition that certain events are disabled.

One possible control objective may be specified as a pair of p-languages $K_1 \preceq K_2$. The "upper" p-language K_2 may be deterministic specifying a legality specification (anything in its complement is considered illegal), whereas the "lower" p-language K_1 imposes restriction on the occurrence probability of the legal traces, which must occur with certain minimum probabilities. The goal is to selectively disable controllable events so that the p-language of the supervised system lies within the range of the prescribed p-languages. It is easy to see that this framework generalizes that of Ramadge-Wonham and reduces to the Ramadge-Wonham framework when all p-languages are deterministic languages.

A Metric for Probabilistic Languages

In this appendix we show that a suitable metric can be defined on the space of p-languages under which it is complete, and the choice and concatenation functions are both continuous and contracting.

Definition 12 A metric $d : \mathcal{L} \times \mathcal{L} \to \mathcal{R}_+$ on the set of p-languages is defined as follows:

$$\forall K, L \in \mathcal{L} : d(K, L) := \sup_s |K(s) - L(s)|.$$

It is well known that d as defined above is a metric, i.e., it is non-negative, symmetric, satisfies triangle inequality, and its value for a p-language pair is zero if and only if the two languages are the same. The following result states that $\mathcal{L}$ is complete under this metric.

Theorem 12 $(\mathcal{L}, d)$ is a complete metric space.

To show the completeness of $(\mathcal{L}, d)$, we show that every Cauchy sequence converges. Consider a Cauchy sequence of p-languages $\{L_i\}_{i\geq 0}$. Since $\mathcal{L}$ is a subspace of real valued functions, which is a complete metric space, the limit of the Cauchy sequence, say L_∞, exists in the space of real valued functions. We need to show that $L_\infty \in \mathcal{L}$.

P1 follows from the fact that:

$$L_\infty(\epsilon) = \lim_{i\to\infty} L_i(\epsilon) = \lim_{i\to\infty} 1 = 1.$$

Next to see P2 we have:

$$\begin{aligned}
\sum_\sigma L_\infty(s\sigma) &= \sum_\sigma \lim_{i\to\infty} L_i(s\sigma) \\
&= \lim_{i\to\infty} \sum_\sigma L_i(s\sigma) \\
&\quad \{\text{applying P2 to } L_i\} \\
&\leq \lim_{i\to\infty} L_i(s) \\
&= L_\infty(s).
\end{aligned}$$

■

In the the following theorem we show that the choice and concatenation functions are both contracting.

Theorem 13 Consider a p-language $L \in \mathcal{L}$ and $e \in [0,1]$. Then the functions $+_e L$, $._e L : \mathcal{L} \to \mathcal{L}$ are both contracting (and hence continuous) whenever $e \neq 0$ and $e \neq 1$, respectively.

Proof: Consider $K_1, K_2 \in \mathcal{L}$. Then we have

$$\begin{aligned} d(+_e L(K_1), +_e L(K_2)) &= \sup_s |(eK_1(s) + \bar{e}L(s)) - (eK_2(s) + \bar{e}L(s))| \\ &= e \sup_s |K_1(s) - K_2(s)| \\ &= ed|K_1, K_2|, \end{aligned}$$

which shows that $+_e L$ is a contraction whenever $e \neq 0$ (so that $\bar{e} \neq 1$). Similarly,

$$\begin{aligned} & d(._e L(K_1), ._e L(K_2)) \\ = \; & \sup_s |(L(s) - e\Delta(L)(s) + e\Delta(L) \circ K_1(s)) - (L(s) - e\Delta(L)(s) + e\Delta(L) \circ K_2(s))| \\ & \{\text{from linearity of convolution}\} \\ = \; & e \sup_s |\Delta(L) \circ (K_1(s) - K_2(s))| \\ = \; & e \sup_s \left| \sum_t \Delta(L)(t)[K_1(t^{-1}s) - K_2(t^{-1}s)] \right| \\ & \{\text{since } K_1(t^{-1}s) - K_2(t^{-1}s) \leq \sup_t |K_1(t^{-1}s) - K_2(t^{-1}s)|\} \\ \leq \; & e \sup_s \left[\sup_t |K_1(t^{-1}s) - K_2(t^{-1}s)| \right] \sum_t \Delta(L)(t) \\ & \{\text{since } \sum_t \Delta(L)(t) \leq 1, \text{ Theorem 1, part 1}\} \\ \leq \; & e \sup_s \sup_t |K_1(t^{-1}s) - K_2(t^{-1}s)| \\ & \{\text{applying change of variables } u := t^{-1}s\} \\ = \; & e \sup_u |K_1(u) - K_2(u)| \\ = \; & ed(K_1, K_2), \end{aligned}$$

which shows that $._e L$ is contraction whenever $e \neq 1$. ■

Remark 8 Since choice and concatenation functions are both contractions defined over the complete metric space $(\mathcal{L}, d)$, it follows from the contraction mapping theorem [19] that they possess *unique* fixed points. (The uniqueness of fixed points is not guaranteed by lattice theoretic approach of Theorems 4 and 6.) The same holds for any function f composed of the choice and the concatenation function. The unique fixed point of f can be obtained by taking the limit of the sequence of p-languages $\{L_i\}$ defined recursively by $L_{i+1} = f(L_i)$, and $L_0 \in \mathcal{L}$ is arbitrary.

References

[1] J. Berstel and C. Reutenauer. *Rational Series and their Languages.* Springer-Verlag, New York, 1988.

[2] C. G. Cassandras. *Discrete Event Systems: Modeling and Performance Analysis.* Aksen Associates, Boston, MA, 1993.

[3] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order.* Cambridge University Press, Cambridge, UK, 1990.

[4] E. Doberkat. *Stochastic Automata: Stability, Nondeterminism and Prediction*, volume 113 of *Lecture Notes in Computer Science.* Springer-Verlag, New York, 1981.

[5] W. Feller. *An Introduction to Probability Theory and Its Applications, Vol. 1.* Wiley, New York, NY, 2nd edition, 1966.

[6] V. K. Garg. An algebraic approach to modeling probabilistic discrete event systems. In *Proceedings of 1992 IEEE Conference on Decision and Control*, pages 2348–2353, Tucson, AZ, December 1992.

[7] V. K. Garg. Probabilistic languages for modeling of deds. In *Proceedings of Conference on Information Sciences and Systems*, pages 198–203, Princeton, NJ, March 1992.

[8] G. Hansel and D. Perrin. Mesures de probabilites rationnelles. Mots, Hermes, 1990.

[9] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, Reading, MA, 1979.

[10] R. Kumar and V. K. Garg. Extremal solutions of inequations over lattices with applications to supervisory control. *Theoretical Computer Science*, 148:67–92, November 1995.

[11] R. Kumar and V. K. Garg. *Modeling and Control of Logical Discrete Event Systems.* Kluwer Academic Publishers, Boston, MA, 1995.

[12] E. T. Lee and L. A. Zadeh. Note on fuzzy languages. *Information Sciences*, pages 421–434, 1969.

[13] M. K. Molloy. Performance analysis using stochastic Petri nets. *IEEE Transactions on Computers*, C-31(9):913–917, September 1982.

[14] H. Moon and W. Kwon. Performance evaluation of a discrete event system via the regular expression of trajectories. In *Proceedings of 1996 International Workshop on Discrete Event Systems*, pages 226–231, Edinburgh, UK, August 1996.

[15] H. Mortazavian. Controlled stochastic languages. In *Proceedings of 1993 Allerton Conference*, pages 938–947, Urbana, IL, 1993.

[16] A. Paz. *Introduction to Probabilistic Automata.* Academic Press, New York, 1971.

[17] M. O. Rabin. Probabilistic automata. *Information and Control*, 6:230–245, 1963.

[18] P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal of Control and Optimization*, 25(1):206–230, 1987.

[19] W. Rudin. *Priciples of Mathematical Analysis.* McGraw-Hill, 1976.

[20] A. Salomaa. Formal languages and power series. In J. v. Leeuwen, editor, *Handbook of theoretical computer science.* MIT Press, Cambridge, MA, 1994.

[21] E. Wong and B. Hajek. *Stochastic processes in engineering systems.* Springer-Verlag, New York, 1985.