# A Formal Mathematical Framework for Modeling Probabilistic Hybrid Systems

**Robert St-Aubin**
*staubin@cs.ubc.ca*

**Joel Friedman**
*jf@cs.ubc.ca*
Department of Computer Science
University of British Columbia
Vancouver, BC V6T 1Z4

**Alan K. Mackworth**
*mack@cs.ubc.ca*

## Abstract

The development of autonomous agents, such as mobile robots and software agents, has generated considerable research in recent years. Robotic systems, which are usually built from a mixture of continuous (analog) and discrete (digital) components, are often referred to as hybrid dynamical systems. Traditional approaches to real-time hybrid systems usually define behaviors purely in terms of determinism or sometimes non-determinism. However, this is insufficient as real-time dynamical systems very often exhibit uncertain behaviour. To address this issue, we develop a semantic model, *Probabilistic Constraint Nets* (PCN), for probabilistic hybrid systems. PCN captures the most general structure of dynamic systems, allowing systems with discrete and continuous time/variables, synchronous as well as asynchronous event structures and uncertain dynamics to be modeled in a unitary framework. Based on a formal mathematical paradigm uniting abstract algebra, topology and measure theory, PCN provides a rigorous formal programming semantics for the design of hybrid real-time embedded systems exhibiting uncertainty.

## 1 Introduction

Dynamical systems are defined on time structures and domain structures. Both of which can be either discrete or continuous. A *hybrid* dynamical system is a dynamical system composed of a combination of discrete and continuous time and domain structures. A robotic system consisting of a computer-controlled robot coupled to a continuous environment is an example of a hybrid dynamical system. Zhang and Mackworth proposed a formal framework for deterministic hybrid systems called *Constraint Nets* (CN) [17]. Although their paradigm allows for the modeling of non-deterministic systems through hidden inputs, it does not permit the specification of uncertainty in the system. However, real-time dynamical systems very often behave unpredictably and thus exhibit (structured) uncertainty. It is therefore important to be able to model and analyze real-time probabilistic systems.

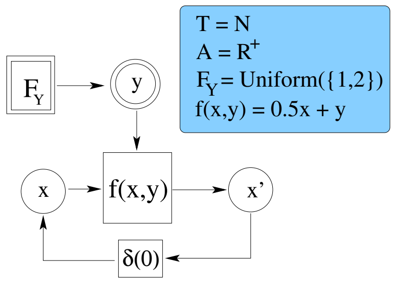


Figure 1: Simple Transduction with uncertainty

In this paper we introduce a sound mathematical framework for the modeling of probabilistic hybrid dynamical systems that we call *Probabilistic Constraint Nets* (PCN). PCN provides a model that is formal and general, modular and composite, powerful and practical. Moreover, PCN has a graphical representation which simplifies the modeling task. Based on algebraic, topological and measure-theoretic structures of dynamics, the PCN framework extends the CN paradigm to allow the user to model uncertainty in the dynamics of the system and in the environment. We will introduce the syntax of the modeling language along with its semantics which leads to a fixpoint in distribution.

However, before introducing the syntax of our framework, we present an example of a basic dynamical system that we will use throughout this paper. Consider the discrete time dynamical system corresponding to the following recursive function: $f(t+1) = 0.5f(t) + Y(\omega), f(0) = 0$, where $Y(\omega) : \Omega \to \{1, 2\}$ is a random variable with a discrete uniform distribution over the set $\{1, 2\}$. The PCN graphical representation of this system is depicted in Figure 1. This system will serve as a running example throughout this paper. Figure 1 will be explained when we introduce the PCN syntax in Section 3. We will also use this example to illustrate the semantics of the PCN framework in Section 4.

## 2 Mathematical Foundations

In this section, we present the essential mathematical concepts needed to understand the theoretical results for the semantics of the PCN framework. We assume the reader is familiar with topology and measure theory concepts that can be found in [2, 3] and in the first author's dissertation [13].

As we are interested in modeling dynamical systems, a model of time and its evolution is necessary. In fact, a clear notion of the concept of time is central to understanding dynamics. We formalize time using an abstract structure that captures its most important properties. In general, a time structure can be considered as a totally ordered set with an initial start time, an associated metric for "the distance between any two time points" and a measure for "the duration of an interval of time." Formally, we define the concept of time structure as follows.

**Definition 2.1 (Time structure)** *A* time structure *is a triple $\langle \mathcal{T}, d, \mu\rangle$*[1] *where*

- *$\mathcal{T}$ is a linearly ordered set $\langle \mathcal{T}, \leq\rangle$ with $\mathbf{0}$ as the least element;*
- *$\langle \mathcal{T}, d\rangle$ forms a metric space with $d$ as a metric satisfying: for all $t_0 \leq t_1 \leq t_2$,*
$$d(t_0, t_2) = d(t_0, t_1) + d(t_1, t_2),$$
*$\{t | m(t) \leq \tau\}$ has a greatest element and $\{t | m(t) \geq \tau\}$ has a least element for all $0 \leq \tau < \sup\{m(t) | t \in \mathcal{T}\}$ where $m(t) = d(\mathbf{0}, t)$;*
- *$\langle \mathcal{T}, \sigma, \mu\rangle$ forms a measure space with $\sigma$ as the Borel set of topological space $\langle \mathcal{T}, d\rangle$ and $\mu$ as a Borel measure satisfying $\mu([t_1, t_2)) \leq d(t_1, t_2)$ for all $t_1 \leq t_2$ where $[t_1, t_2) = \{t | t_1 \leq t < t_2\}$ and $\mu([t_1, t_2)) = \mu([\mathbf{0}, t_2)) - \mu([\mathbf{0}, t_1))$.*

As with time, we formalize domains as abstract structures so that discrete and continuous domains are defined uniformly. A domain can be either simple or composite. Simple domains denote simple data types, such as reals, integers, Booleans and characters; composite domains denote structured data types, such as arrays, vectors, strings, objects, structures and records.

**Definition 2.2 (Simple domain)** *A* simple domain *is a pair $\langle A \cup \{\perp_A\}, d_A\rangle$ where $A$ is a set, $\perp_A \notin A$ means undefined in A, and $d_A$ is a metric on A.*

Let $\overline{A} = A \cup \{\perp_A\}$. For simplicity, we will use $\overline{A}$ to refer to simple domain $\langle \overline{A}, d_A\rangle$ when no ambiguity arises. For example, let $\mathbb{R}$ be the set of real numbers, $\overline{\mathbb{R}}$ is a simple domain with a connected metric space; let $\mathbb{B} = \{0, 1\}$, $\overline{\mathbb{B}}$ is a simple domain with a discrete topology on $\mathbb{B}$.

Any simple domain $\overline{A}$ is associated with a partial order relation $\leq_{\overline{A}}$. $\langle \overline{A}, \leq_{\overline{A}}\rangle$ is a flat partial order with $\perp_A$ as the least element. In addition, $\overline{A}$ is associated with a *derived metric topology* $\tau = \tau_A \cup \{\overline{A}\}$ where $\tau_A$ is the metric topology on $A$ derived from the metric $d_A$.

[1]To abbreviate the notation, we will simply use $\mathcal{T}$ to refer to the time structure $\langle \mathcal{T}, d, \mu\rangle$ when no ambiguity arises.

A domain is defined recursively based on simple domains.

**Definition 2.3 (Domain)** *$\langle A, \leq_A, \tau\rangle$, with $\leq_A$ as the partial order relation and $\tau$ as the derived metric topology, is a* domain[2] *iff:*

- *it is a simple domain; or*
- *it is a* composite *domain, i.e., it is the product of a family of domains $\{\langle A_i, \leq_{A_i}, \tau_i\rangle\}_{i\in I}$ such that $\langle A, \leq_A\rangle$ is the product partial order of the family of partial orders $\{\langle A_i, \leq_{A_i}\rangle\}_{i\in I}$ and $\langle A, \tau\rangle$ is the product space of the family of topological spaces $\{\langle A_i, \tau_i\rangle\}_{i\in I}$.*

We take a signature as a syntactical structure of a class of multi-sorted domains with associated functions defined on these domains. Let $\Sigma = \langle S, F\rangle$ be a *signature* where $S$ is a set of *sorts* and $F$ is a set of *function symbols*. $F$ is equipped with a *mapping type*: $F \to S^* \times S$ where $S^*$ denotes the set of all finite tuples of $S$. For any $f \in F$, $type(f)$ is the type of $f$. We use $f : s^* \to s$ to denote $f \in F$ with $type(f) = \langle s^*, s\rangle$. For example, the signature of an algebra on the Naturals can be denoted by $\Sigma_{\mathbb{N}} = \langle \mathbb{N}, \{0, +, -, \times\}\rangle$. This signature has only one sort, $\mathbb{N}$, with 4 different function symbols.

A domain structure of a signature is defined as follows. Let $\Sigma = \langle S, F\rangle$ be a signature. A $\Sigma$-*domain structure* $A$ is a pair $\langle \{A_s\}_{s\in S}, \{f^A\}_{f\in F}\rangle$ where for each $s \in S$, $A_s$ is a domain of sort $s$, and for each $f : s^* \to s \in F$ with $s^* : I \to S$ and $s \in S$, $f^A : \times_I A_{s_i^*} \to A_s$ is a function denoted by $f$, which is continuous in the partial order topology. For example, $\langle \overline{\mathbb{N}}, \{0, +, -, \times\}\rangle$ is a $\Sigma_{\mathbb{N}}$ structure where $+$, $-$ and $\times$ are addition, subtraction and multiplication, respectively.

With any time structure and domain structure, we can define two basic elements in probabilistic dynamical systems: stochastic traces, which are functions of time and sample space $\Omega$, and transductions, which are mappings from stochastic traces to stochastic traces.

Stochastic traces are a central notion in representing the dynamical behaviour of the systems modeled within the PCN framework. A stochastic trace intuitively denotes the (random) changes of values over time. Formally, a *stochastic trace* is a mapping $v : \Omega \times \mathcal{T} \to A$ from sample space $\Omega$ and time domain $\mathcal{T}$ to value domain $A$. For a given $\omega \in \Omega$, the function $v_\omega : \mathcal{T} \to A$ is simply called a *trace*. In the literature, a trace is often referred to as a sample function, a realization, a trajectory or a path of the underlying stochastic process. We will use $v$ to denote both the stochastic trace $v$ or one of its realization traces $v_\omega$ when it is clear from the context and no ambiguity arises.

A stochastic trace $v$ is *well-defined* iff $v(\omega, t)$ is well-defined for all $(\omega, t) \in \Omega \times \mathcal{T}$. A stochastic trace $v$ is *undefined* iff $v(\omega, t)$ is undefined for any $(\omega, t) \in \Omega \times \mathcal{T}$. For example, denote a Brownian motion process by $B_t(\omega)$

[2]For simplicity, we will use $A$ to refer to domain $\langle A, \leq_A, \tau\rangle$ when no ambiguity arises.

and $\mathcal{T} = \mathbb{R}^+$ and $A = \overline{\mathbb{R}}$. Then $v = \lambda\omega, t.B_t(\omega)$ is a well-defined stochastic trace. For a fixed $\omega$ in $\Omega$, $v_\omega = \lambda t.B_t(\omega)$ represents a path of the Brownian motion process. On the other hand, $v_1 = \lambda t.\cos(t)$ and $v_2 = \lambda t.e^{-t}$ are well-defined deterministic traces, i.e., stochastic traces for which $|\Omega| = 1$.

Due to the fact that physical systems encompass uncertainty, one is often more interested in the distribution of the set of all execution traces of system rather than in one specific execution trace.

One important feature of a trace is that it provides complete information about the current execution of the system of interest at every time point. In the presence of uncertainty, the limiting value of a specific execution trace $v_\omega$ is of little interest since the measure of that trace is typically zero. The distribution of a stochastic trace, on the other hand, provides complete information about the probability of the state of the system at every finite time point.

A transduction is a causal mapping from input stochastic traces to output stochastic traces. The causal relationship stipulates that the evolution of the system cannot be dictated by the future state of the system, but only by past and present values. Formally, causality can be defined as follows

**Definition 2.4 (Causality via $\mathcal{F}_t$-adaptedness)** *Assume $\{\mathcal{F}_t\}_{t\geq 0}$ to be an increasing family of $\sigma$-algebra of subsets of $A^{\Omega\times\mathcal{T}}$. A mapping $F(v)(\omega,t) : A^{\Omega\times\mathcal{T}} \rightarrow A'^{\Omega\times\mathcal{T}'}$ is* causal *if $F(v)(\omega,t)$ is $\mathcal{F}_t$-adapted. A causal mapping on stochastic trace spaces is called a* transduction.

Primitive transductions are defined on a generic time structure $\mathcal{T}$ and are functional compositions of three types of basic transduction: *generators*, *transliterations* and *delays*.

**Definition 2.5 (Generator)** *Let $A$ be a domain, $\Omega$ be a sample space and $\mathcal{T}$ a time structure. Moreover, let $F_{X|A}$ denote the (potentially conditional) cumulative distribution function for the random variable $X$. A* generator *$\mathscr{G}^A_{\mathcal{T}}(v_0) : \Omega \times \mathcal{T} \times A \rightarrow A$ is a basic transduction defined as*

$$\mathscr{G}^A_{\mathcal{T}}(v_0, F_X)(v) = \lambda\omega, t. \begin{cases} v_0 & \textit{if } t = 0 \\ rand(F_{X|v(\omega,t)}(t), \omega) & \textit{else} \end{cases}$$

*where $rand(F_{X|A}, \cdot)$ is a random number generator associated with $F_{X|A}$.*

We allow the distribution function $F_{X|A}$ to be conditioned on $t$ and values of the systems to produce a general model of uncertainty. This enables the user to model systems where the uncertainty component is non-stationary and conditioned on the state of the system. Also note that in this paper, we are not interested in the simulation of random variables per se, but rather in the analysis of the resulting models. Hence, we will assume that we are given, for each generator included in the model, appropriate random number generators [5, 6].

**Definition 2.6 (Transliteration)** *A* transliteration *is a pointwise extension of a function. Formally, let $f : \Omega \times A \rightarrow A'$ be a function and $\mathcal{T}$ be a time structure. The pointwise extension of $f$ onto $\mathcal{T}$ is a mapping $f_{\mathcal{T}} : A^{\Omega\times\mathcal{T}} \rightarrow A'^{\Omega\times\mathcal{T}}$ satisfying $f_{\mathcal{T}}(v) = \lambda\omega, t.f(v(\omega,t))$.*

By this definition, $(f \circ g)_{\mathcal{T}} = f_{\mathcal{T}} \circ g_{\mathcal{T}}$. We will also use $f$ to denote transliteration $f_{\mathcal{T}}$ if no ambiguity arises.

Intuitively, a transliteration is a transformational process without memory or internal state, such as a combinational circuit. Note that in the absence of any random variable within the transliteration, the transformational process is simply a deterministic function of the current input.

Now let us present the last type of basic transduction: delays. There are two types of delay: unit delays and transport delays.

For a given trace, a unit delay $\delta^A_{\mathcal{T}}(\omega, v_0)$ acts as a unit memory for data in domain $A$, given a discrete time structure. We will use $\delta(v_0)$ to denote unit delay $\delta^A_{\mathcal{T}}(\omega, v_0)$ if no ambiguity arises.

**Definition 2.7 (Unit delay)** *Let $A$ be a domain, $v_0$ a well-defined value in $A$, and $\mathcal{T}$ a discrete time structure. A* unit delay *$\delta^A_{\mathcal{T}}(\omega, v_0) : A^{\Omega\times\mathcal{T}} \rightarrow A^{\Omega\times\mathcal{T}}$ is a transduction defined as*

$$\delta^A_{\mathcal{T}}(\omega, v_0)(v) = \lambda t. \begin{cases} v_0 & \textit{if } t = \mathbf{0} \\ v(\omega, pre(t)) & \textit{otherwise} \end{cases}$$

*where $v_0$ is called the* initial output value *of the unit delay.*

However, in the presence of non-discrete time structures, unit delays may not be meaningful. Hence we need a transduction that is suitable for more general time structures.

**Definition 2.8 (Transport delay)** *Let $A$ be a domain, $v_0$ a well-defined value in $A$, $\mathcal{T}$ a time structure and $\tau > 0$. A* transport delay *$\Delta^A_{\mathcal{T}}(\tau)(\omega, v_0) : A^{\Omega\times\mathcal{T}} \rightarrow A^{\Omega\times\mathcal{T}}$ is a transduction defined as*

$$\Delta^A_{\mathcal{T}}(\tau)(\omega, v_0)(v) = \lambda t. \begin{cases} v_0 & \textit{if } m(t) < \tau \\ v(\omega, t - \tau) & \textit{otherwise} \end{cases}$$

*where $v_0$ is called the* initial output value *of the transport delay and $\tau$ is called the* time delay.

We will use $\Delta(\tau)(v_0)$ to denote transport delay $\Delta^A_{\mathcal{T}}(\tau)(\omega, v_0)$ if no ambiguity arises. Transport delays are essential for modeling sequential behaviors in dynamical systems.

With preliminaries established, we define an abstract structure of dynamics.

**Definition 2.9 ($\Sigma$-dynamics structure)** *Let $\Sigma = \langle S, F\rangle$ be a signature. Given a $\Sigma$-domain structure $A$ and a time structure $\mathcal{T}$, a* $\Sigma$-dynamics structure *$\mathcal{D}(\mathcal{T}, A)$ is a pair $\langle \mathcal{V}, \mathcal{F}\rangle$ such that*

- *$\mathcal{V} = \{A_s^{\Omega\times\mathcal{T}}\}_{s\in S} \cup \mathcal{E}^{\Omega\times\mathcal{T}}$ where $A_s^{\Omega\times\mathcal{T}}$ is a stochastic trace space of sort $s$ and $\mathcal{E}^{\Omega\times\mathcal{T}}$ is the stochastic event space;*
- *$\mathcal{F} = \mathcal{F}_{\mathcal{T}} \cup \mathcal{F}^\circ_{\mathcal{T}}$ where $\mathcal{F}_{\mathcal{T}}$ is the set of basic transductions, including the set of transliterations $\{f^A_{\mathcal{T}}\}_{f\in F}$, the set of unit delays $\{\delta^{A_s}_{\mathcal{T}}(v_s)\}_{s\in S, v_s\in A_s}$, the set of transport delays $\{\Delta^{A_s}_{\mathcal{T}}(\tau)(v_s)\}_{s\in S,\tau>0,v_s\in A_s}$, and the set of generators $\{\mathscr{G}^{A_s}_{\mathcal{T}}\}_{s\in S}$; $\mathcal{F}^\circ_{\mathcal{T}}$ is the set of event-driven transductions derived from the set of basic transductions, i.e., $\{F^\circ | F \in \mathcal{F}_{\mathcal{T}}\}$.*

## 3 Syntax of PCN

A probabilistic constraint net consists of a finite set of locations, a finite set of transductions and a finite set of connections. However, in order to be able to handle the uncertainty in the systems that we model, we add an essential component: the *generator*. A generator acts as a random number generator, following a given probability distribution and inducing a random location as its output. Thus, in practice, generators can be represented as discrete (e.g. Poisson, uniform) or continuous (Gaussian, exponential) probability distributions although we will use a general (and formal) measure theoretic definition.

**Definition 3.1 (Probabilistic Constraint Nets)** *A probabilistic constraint net is a tuple $PCN = \langle Lc, Td, Cn\rangle$, where $Lc$ is a finite set of locations, each associated with a sort; $Td$ is a finite set of labels of transductions (either deterministic or probabilistic), each with an output port and a set of input ports, and each port is associated with a sort; $Cn$ is a set of connections between locations and ports of the same sort, with the restrictions that (1) no location is isolated, (2) there is at most one output port connected to each location, (3) each port of a transduction connects to a unique location.*

Intuitively, each location is of fixed sort; a location's value typically changes over time. A location can be regarded as a wire, a channel, a variable, or a memory cell. An output location of a generator will be viewed as a *random* variable.

Each transduction is a causal mapping from inputs to output over time, operating according to a certain reference time or activated by external events. Note that probabilistic transductions are built of at least one basic generator transduction. Every generator is associated with a given probability distribution, either discrete or continuous, thus the sort of the output of a probabilistic transduction is the sort of its probability distribution.

Connections link locations with ports of transductions. A clock is a special kind of location connected to the input event port of event-driven transductions. We will introduce the notion of event-driven transduction in Section 5.

A location $l$ is called an *output* location of a PCN iff $l$ connects to the output port of a transduction in $Td$; otherwise, since isolated locations are not allowed it is an *input* location. We will use the notation $I(PCN)$ and $O(PCN)$ to denote the set of input and output locations of a probabilistic constraint net $PCN$. A probabilistic constraint net is *open* if there exists at least one input location, otherwise it is said to be *closed*.

Another feature of our framework is its graphical representation. A PCN can be represented by a bipartite graph where locations are depicted by circles, transductions by boxes, generators by double boxes and connections by arcs. To differentiate them from deterministic locations, we depict random locations with double circles.

Most commonly used families of probability distributions are parameterized, i.e., one can fully specify a probability distribution by giving values to the parameters of the family. The ability of generators to be dependent on certain locations of the model also greatly simplifies the design task when modeling a complex system for which the various uncertain inputs are not fully known. Indeed, specifying the parameters of a probability distribution is often hard and counter-intuitive. Therefore, a designer could set the parameters of the distribution to some *default* location value, and then, as the system evolves, learn the values of the parameters of the distribution, thus updating their values as a better estimate is being learned. For example, to model sensor noise with a PCN generator following a Gaussian probability distribution on the discrete time structure $\mathcal{T} = \mathbb{N}$, one would simply need to connect the inputs of the generator to the locations holding the static values of the mean $\mu$ and the variance $\sigma^2$ to generate samples from the Gaussian distribution at every time point in $\mathcal{T}$ (see Figure 2).

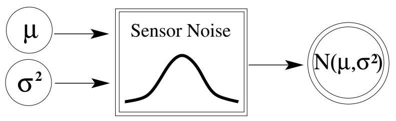


Figure 2: Gaussian probability distribution as a Generator and random location.

To exemplify the graphical syntax of PCN further, let us return to the PCN of Figure 1. In this PCN model, there are three locations ($x'$, $x$ and $y$), one transduction, one generator and one unit delay. The transduction $f(x, y)$ is a transliteration with two inputs, namely $x$ and $y$. The unit delay $\delta(0)$ is introduced to eliminate an algebraic loop and the generator $F_y$ follows a discrete uniform distribution over the set $\{1, 2\}$. Hence, the output of the transduction would be a random sequence of values where the value at time $t + 1$ would be half of the value at time $t$ added to either 1 or 2, with equal probability. The set of possible execution traces resulting from this transduction on $\mathcal{T} = \mathbb{N}$ is $\{0, 1, 2.5, 3.25, 2.625, \ldots\}$. This set of traces has a measure of $0.0625$.

## 4 Semantics of PCN

We have briefly introduced the syntax of the probabilistic constraint nets model, which has the useful properties of being graphical and modular. However, the syntax does not provide a meaning for the model. Indeed, there are multiple models with similar syntax to probabilistic constraint nets (Petri Nets [11] and their generalization Coloured Petri Nets [4] for example) that have completely different interpretations. Therefore, it is necessary to have a formal semantics of probabilistic constraint nets in order to correctly interpret models of complex physical systems.

The fixpoint theory of partial order has been used as a semantical model for programming languages and models [3]: in this case, a program (or a model) defines a function $f$ and its semantics are defined to be the least solution of $x = f(x)$, or the least fixpoint of $f$. A similar approach was developed to provide a fixpoint semantics for the Constraint Net model [17]. However, even though our framework is similar to that of Constraint Nets, the seman-

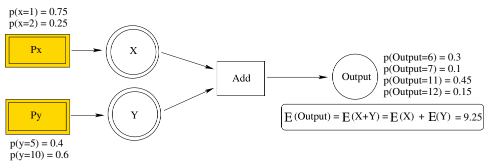
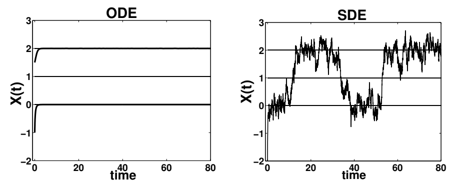


Figure 3: Simple PCN for a probabilistic sum.



(a) (b)

Figure 4: (a) ODE: $\dot{X}_t = -X_t(X_t - 1)(X_t - 2)$;$X_0 = -1$ and $X_0 = 1.5$
(b) SDE: $\dot{X}_t = -X_t(X_t - 1)(X_t - 2) + N_t$; $X_0 = -2$

tics of PCN differ significantly from that of CN, because we have introduced uncertainty into the set of equations induced by the PCN model. Hence, a probabilistic constraint net is a set of equations with locations serving as variables. Some of the variables (locations) in the equations, those that are outputs of generators, are in fact random variables, obeying some probability distribution, which in turn affect the value of the transductions for which they are inputs. Transductions play the role of functions and the connections between locations and transductions generate a set of equations. Obviously, the semantics of a PCN should be a solution to this set of equations containing random variables. Figure 3 demonstrates the effect of random locations on the transductions. Transduction *Add* is a very simple transliteration representing the sum of two (probabilistic) inputs $X$ and $Y$. It is easy to notice that the output value for this transliteration also follows a probability distribution. In this case, there are 4 possible values which each have different likelihood of occurrence. One should note that although the distribution of a random variable is helpful in reasoning about its behaviour, one can reason about statistics such as the expected value, that is, one can redefine the notion of behavior in terms of *average* behavior for the system. In our simple example, we can see that the average output value of the system is 9.25.

Since the equations in a PCN model do not converge to a fixpoint but rather to a stationary distribution, the fixpoint theory of partial order cannot be utilized directly to provide a denotational semantics for PCN. In fact, in the presence of uncertainty in the system, the least solution of an equation with random variables is a Markov stochastic process [8].

To further illustrate the difference between the semantics of a deterministic system (CN) and one encompassing uncertainty (PCN), let us compare two dynamical systems with nominal component

$$\dot{X}_t = -X_t(X_t - 1)(X_t - 2).$$

The first one is deterministic and has two distinct stable at-

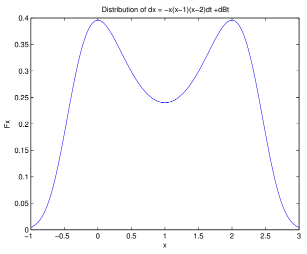


Figure 5: Density of $dX = -X(X-1)(X-2)dt + dB_t$.

tractors (equilibria),[3] at 2 and at 0, as shown in Figure 4(a). The behaviour of this system is fully determined by its initial value and it reaches one of the two stable fixpoints based on this initial value.

The second system, which cannot be modeled with a deterministic constraint net, is stochastically affected by a simple Brownian motion process. A sample path for this system, for an initial value of $X_0 = -2$, is shown in Figure 4(b). For this specific realization, the system is initially attracted toward the closest equilibrium which is at $X = 0$. The system then fluctuates around this attractor, reacting under the influence of the Brownian motion component and, around time $t = 12$, a large enough noise disturbance pushes the system over the value of 1, causing the system to be attracted toward the other equilibrium, at $X = 2$. Another spike of noise flips the system back to the lower equilibrium at $t = 35$ and so on. This example shows the effect of uncertainty on the system and its behaviour.

In this case, there is no fixpoint for this realization nor for the full system. For a set of sample paths with non-zero measure, the system will keep moving back and forth between the two stable equilibria as it is affected by the noise introduced by the Brownian motion component of the equation. However, the system will reach a stationary distribution. That is, in the long run, the probability distribution of the system will remain unchanged, independent of time. The corresponding density function for this distribution is shown in Figure 5. One can clearly observe that the system is symmetrically distributed with higher weight around the two stable equilibria located at $X = 0$ and $X = 2$. One should note that if the effect of the Brownian noise is diminished, the peaks at $X = 0$ and $X = 2$ rise or fall (depending on the starting value) as the noise is less likely to cause a jump large enough to cause the other equilibrium to become the main attractor. Letting the effect of the noise converge to zero would lead to the deterministic case as presented in Figure 4(a), that is, the stationary distribution would be degenerate everywhere except at the equilibrium corresponding to the initial value of the system. Hence a deterministic system is in fact a special case of the more general stochastic system.

We define the semantics for the Probabilistic Constraint

[3]There are in fact three different equilibria, at 0, 1 and 2 respectively. However, the equilibrium at 1 is unstable. Any shift in value will cause the system to move away from this unstable equilibrium and move towards one of the other two stable equilibria.

Net model to be the least fixpoint of the distribution of the solution to the set of equations of the PCN model. These semantics are, as it was mentioned in the previous paragraph, applicable to any system, whether it be stochastic or deterministic.

### 4.1 Fixpoint in distribution of partial orders

The fixpoint theorems used here are for *complete partial orders* (cpo's). *Continuous* functions are functions which are continuous in partial order topologies. A fixpoint in the distribution of a function $f$ can be considered as a solution of the equation $x = f(x)$, where $f(\cdot)$ is an stochastic function. The least fixpoint is the least element in the fixpoint set.

**Definition 4.1 (Fixpoint in distribution and Least fixpoint)** *Let $f : \Omega \times A \to A$ be a function on a sample space $\Omega$ and a partial order $A$. A function $g : \Omega \times A \to A$ is a* fixpoint in distribution *of $f$ iff the distribution of $g$ is a stationary distribution for $f$. It is the* least fixpoint *in distribution of $f$ iff, in addition, $F_g \leq F_{g'}$ for every other function $g'$ which is a fixpoint in distribution of $f$.*

Least fixpoints in distribution, if they exist, are unique. The least fixpoint in distribution of $f$ will be denoted by $\mu.F_f$.

Based on the above definition, we can state our first fixpoint in distribution theorem as follows.

**Theorem 4.1 (Fixpoint Theorem I)** *Let $A$ be a cpo and assume that either $A$ is also a* total order *or that the set of distributions over $A$ is a cpo and the function over distributions is continuous. Then, every continuous function $f : \Omega \times A \to A$ or pathwise continuous function $f_\omega : A \to A$ (for a fixed $\omega \in \Omega$) has a least fixpoint in distribution.*

We now present our second fixpoint in distribution theorem which is applicable to a function of two arguments.

**Theorem 4.2 (Fixpoint Theorem II)** *Let $A$ and $A'$ be two cpos and assume that either $A, A'$ are also* total orders *or that the set of distributions over $A'$ is a cpo and the function over distributions is continuous. If $f : \Omega \times A \times A' \to A'$ is a continuous function, then there exists a unique continuous function $\mu.f : \Omega \times A \to A'$, such that for all $a \in A$, the distribution of $(\mu.f)(a)$ is the least fixpoint in distribution of $\lambda\omega, x.f_\omega(a, x)$.*

Formally, a set of equations can also be written as $\vec{o} = \vec{f}(\vec{\omega}, \vec{i}, \vec{o})$ where $\vec{i}$ is a tuple of input variables and $\vec{o}$ is a tuple of output variables. Based on our previous results, if $\vec{f}$ is continuous, then its least fixpoint in distribution is a continuous function, denoted $\mu.\vec{f}$.

### 4.2 Semantics of Probabilistic Constraint Nets

In this section, we define the fixpoint in distribution semantics of probabilistic constraint nets. Let $\Sigma = \langle S, F \rangle$ be a signature and $c \in S$ be a special sort for clocks. A probabilistic constraint net with signature $\Sigma$ is a tuple $PCN_\Sigma = \langle Lc, Td, Cn \rangle$ where

- each location $l \in Lc$ is associated with a sort $s \in S$, the sort of location $l$ is written as $s_l$;

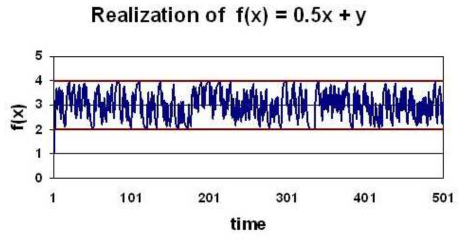


Figure 6: Sample path of the system $f(x) = 0.5x + y$.

- each transduction $F \in Td$ is a basic transduction or an event-driven transduction, the sorts of the input and output ports of $F$ are as follows:
  1. if $F$ is a transliteration of a function $f : s^* \to s \in F$, the sort of the output port is $s$ and the sort of the input port $i$ is $s^*(i)$;
  2. if $F$ is a unit delay $\delta^s$ or a transport delay $\Delta^s$, the sort of both input and output ports is $s$;
  3. if $F$ is an event-driven transduction, the sort of the event input port is $c$, the sorts of the other ports are the same as its primitive transduction;

Let $\mathcal{D}(\mathcal{T}, A) = \langle \mathcal{V}, \mathcal{F} \rangle$ be a $\Sigma$-dynamics structure. $PCN_\Sigma$ on $\langle \mathcal{V}, \mathcal{F} \rangle$ denotes a set of equations $\{o = F_o(\vec{x})\}_{o \in O(PCN)}$, such that for any output location $o \in O(PCN)$,

- $F_o$ is a continuous or pathwise continuous transduction in $\mathcal{F}$ whose output port connects to $o$,
- $\vec{x}$ is the tuple of input locations of $F_o$, i.e., the input port $i$ of $F_o$ connects to location $\vec{x}(i)$.

The semantics of a probabilistic constraint net is defined as follows.

**Definition 4.2 (Semantics)** *The* semantics *of a probabilistic constraint net $PCN$ on a dynamics structure $\langle \mathcal{V}, \mathcal{F} \rangle$, denoted $[\![PCN]\!]$, is the least stationary distribution of the set of equations $\{o = F_o(\vec{x})\}_{o \in O(PCN)}$, given that $F_o$ is a continuous or pathwise continuous transduction in $\mathcal{F}$ for all $o \in O(PCN)$; it is a continuous or pathwise continuous transduction from the input trace space to the output trace space, i.e., $[\![PCN]\!] : \times_{I(PCN)} A_{s_i}^{\Omega \times \mathcal{T}} \to \times_{O(PCN)} A_{s_o}^{\Omega \times \mathcal{T}}$.*

Given any set of output locations $O$, the restriction of $[\![PCN]\!]$ onto $O$, denoted $[\![PCN]\!]_{|O} : \times_{I(PCN)} A_{s_i}^{\mathcal{T}} \to \times_O A_{s_o}^{\mathcal{T}}$, is called *the semantics of PCN for O*. For example, consider the probabilistic constraint net denoted by equations $x' = f(x, \omega) = 0.5x + y(\omega)$ and $x = \delta(0)(x)$ with $F_Y = Uniform(\{1, 2\})$ and $\Omega = \{\omega_1, \omega_2\}$. Given a discrete time structure $\mathbb{N}$, a domain $\overline{\mathcal{I}} = \overline{\{1, 2\}}$ for inputs and a domain $\overline{\mathcal{O}} = \overline{\mathbb{R}}$ for output, the semantics for $x$ is $F : \overline{\mathcal{I}}^{\Omega \times \mathbb{N}} \to \overline{\mathbb{R}}^{\Omega \times \mathbb{N}}$ such that $F(v)(0) = 0$ and $F(v)(n) = f(F(v)(n-1), v(n-1))$ where the limiting distribution for $F$ is stationary.

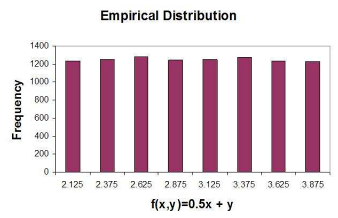


Figure 7: Empirical Distribution of $f(x) = 0.5x + y$ after 10000 time steps.

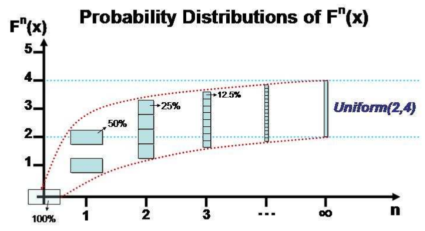


Figure 8: Evolution of the distributions of f(x).

Let us show the derivation of the semantics of this model. In Figure 6, we plot a realization trace of the system, while in Figure 7 we can see the empirical distribution of the system after 10000 time steps. The least fixpoint distribution follows a uniform distribution over the range $[2, 4]$. The evolution of the distributions is presented in Figure 8. One can see that the system's distribution starts as uniform over the range $\{1, 2\}$ and the distribution gradually increases to reach a stationary distribution which follows a uniform distribution over $[2, 4]$.

## 5 Modeling in PCN

We are interested in modeling the larger class of hybrid probabilistic dynamical systems, that is, systems encompassing components of more than one basic type. Within the PCN paradigm, a probabilistic hybrid dynamical system consists of modules with different time structures, with its domain structure multi-sorted and with a set of probabilistic generators, as basic transductions, which allows for the modeling of the uncertain components of these modules.

To model systems with modules that are associated with different clocks we introduce the notion of *event-driven* transductions. In order to properly introduce the notion of event-driven transductions, we need to define the concept of sample and extension traces. Let $\mathcal{T}_r$ be a reference time of $\mathcal{T}$ with a reference time mapping $h$. The *sample stochastic trace* of $v : \Omega \times \mathcal{T}_r \to A$ onto $\mathcal{T}$ is a stochastic trace $\underline{v} : \Omega \times \mathcal{T} \to A$ satisfying $\underline{v} = \lambda\omega, t.v(\omega, h(t))$. The *extension stochastic trace* of $v : \Omega \times \mathcal{T} \to A$ onto $\mathcal{T}_r$ is a stochastic trace $\overline{v} : \Omega \times \mathcal{T}_r \to A$ satisfying

$$\overline{v} = \lambda\omega, t_r. \begin{cases} v(\omega, h^{-1}(t_r)) & \text{if cond} \\ \bot_A & \text{otherwise} \end{cases}$$

where cond $= \exists t \in \mathcal{T}, \mu_r([\mathbf{0}_r, t_r)) \leq \mu([\mathbf{0}, t))$ or $\mu_r([\mathbf{0}_r, t_r)) < \mu(\mathcal{T})$ and $h^{-1}(t_r) = \{t | h(t) \leq_r t_r\} \in \mathcal{T}^\infty$.

Both sampling and extension can be seen as transformational processes on traces, hence they are transductions. *Sampling* is a transduction whose output is a sample trace of its input. *Extending* is a transduction whose output is an extension trace of its input.

An event-driven transduction is a primitive transduction augmented with an extra input which is an event trace; it operates at each event point and the output value holds between two events. This additional event trace input of an event-driven transduction is called the *clock* of the transduction. Intuitively, an event-driven transduction works as follows. First, the input trace with the reference time $\mathcal{T}$ is sampled onto the sample time $\mathcal{T}_e$ generated by the event trace $e$. Then, the primitive transduction is performed on $\mathcal{T}_e$. Finally, the output trace is extended from $\mathcal{T}_e$ back to $\mathcal{T}$.

**Definition 5.1 (Event-driven transduction)** *Let $\mathcal{T}$ be a time structure and $F_\mathcal{T} : A^{\Omega\times\mathcal{T}} \to A'^{\Omega\times\mathcal{T}}$ a primitive transduction. Let $\mathcal{E}^{\Omega\times\mathcal{T}}$ be the set of all stochastic event traces on time structure $\mathcal{T}$. The* event-driven transduction *of F is a mapping $F^\circ_\mathcal{T} : \mathcal{E}^{\Omega\times\mathcal{T}} \times A^{\Omega\times\mathcal{T}} \to A'^{\Omega\times\mathcal{T}}$ satisfying:*

$$F^\circ_\mathcal{T}(e, v) = \begin{cases} \lambda t.\ \bot_{A'} & \textit{if } e = \lambda t.\ \bot_\mathcal{B} \\ \overline{F_{\mathcal{T}_e}(\underline{v})} & \textit{otherwise.} \end{cases}$$

We will use $F^\circ$ to denote event-driven transduction $F^\circ_\mathcal{T}$ if no ambiguity arises.

Hence, we can unify, within the same model, modules with different sample time structures generated by event traces. There are two ways in which an event trace can be generated: either with a fixed sampling rate, or by an event generator that reacts to changes in its inputs. Moreover, we can also combine multiple event traces, yielding new event traces. Typically, event traces are combined using event logic which allow various asynchronous components within a given set of modules to be coordinated. Common logical interactions are “event or", “event and", and “event select". With event logic modules, asynchronous components can be coordinated.

We have modeled and analyzed several real world applications within the PCN framework. Such applications include an elevator system with uncertain passenger arrivals, a museum surveillance robot and a package delivery robot. The models and analysis can be found elsewhere [13, 14, 15]. Due to the hybrid nature of these system, their modeling require generality which is attained via the unitary model proposed within PCN. Although it is possible to model such systems by combining various frameworks, for example by combining finite state machines and PID controllers, it could lead to unclear semantics which would be a serious shortcoming of that approach.

## 6 Related Work and Conclusion

The motivation for developing the PCN framework was to be able to model hybrid dynamical systems while considering the underlying uncertainty in the system. Uncertainty is inherent in any physical system, hence modeling its effects and considering its impact on system behaviour is essential. While development of models for hybrids systems has been very active in the last few years [7, 10], there also exist a multitude of paradigms that allow the modeling of uncertain system. Such paradigms include Markov Processes [8], Markov Decision Processes [12] and Dynamic Bayesian Networks [9]. However, in most cases, these models are either hybrid and deterministic, or stochastic and restricted to a single time structure (either discrete or continuous). We have shown, in the dissertation associated with this work [13], that PCN subsumes most existing computational models handling uncertainty, and that hybrid, sequential and analog computations can be modeled effectively. The advantages of the subsumption offered by PCN, other than the obvious advantage of parsimony, are many. They include ease of implementation, absence of redundancy while avoiding the requirement for the system designer to have to learn and master multiple paradigms.

In conclusion, we have developed a semantic model for uncertain hybrid dynamical systems, that we call Probabilistic Constraint Nets (PCN). Based on abstract algebra, topology and measure theory, we have represented both time and domains in abstract forms, and uniformly formalized basic elements of dynamical systems in terms of traces, transductions and probabilistic transductions. Furthermore, we have also studied both primitive and event-driven transductions which are important elements of dynamical systems, with or without uncertainty.

Since PCN is an abstraction and generalization of dataflow networks, with the addition that we explicitly handle the uncertain components of the system. Within this framework, the behaviour of a system (the semantics of a PCN model) is formally obtained using both the theory of continuous algebra and stochastic systems. Specifically, a probabilistic constraint net models an uncertain dynamical system as a set of interconnected transductions, while the behaviour of the system is the set of input/output traces of the system satisfying all the relationships (constraints on the dynamics) imposed by the transductions. PCN models a hybrid system using event-driven transductions, while the events are generated and synchronized within the system.

Complementary work on PCN was performed and led to the development of language specification for behavioural constraints on the dynamics of the systems. Moreover, verification techniques were also developed to allow for the probabilistic verification of the behavioural constraints [15]. A control synthesis approach was also developed which enables the system designer to synthesize the controller component of a PCN model, hence simplifying the modeling task greatly [13].

## 7 Acknowledgments

The authors gratefully acknowledge discussions with Ying Zhang. This work is supported by the Natural Sciences and Engineering Research Council and Precarn Inc.

## References

[1] G. Cassela and R. Berger. *Statistical Inference*. Belmont, California: Wadsworth and Brooks/Cole, 1990.

[2] Micheal C. Gemignani. *Elementary topology*. Addison-Wesley, 1967.

[3] M. Hennessy. *Algebraic Theory of Processes*. MIT Press, 1988.

[4] K. Jensen. Coloured petri nets and the invariant-method. *Theor. Comp. Science 14*, pages 317–336, 1981.

[5] A.M. Law and W.D. Kelton. *Simulation Modeling and Analysis*. McGRaw-Hill, New York, 3rd edition, 2000.

[6] P. L'Ecuyer. *Handbook of Simulation*, chapter 4: Random Number Generation, pages 93–137. Wiley, 1998.

[7] O. Maler, Z. Manna, and A. Pnueli. From timed to hybrid systems. *Real-Time: Theory in practice in Lecture Notes in Computer Science*, pages 448–484, 1992.

[8] G. Maruyama. Continuous markov processes and stochastic equations. *Rend. Circ. Mat. Palermo*, 4:48–90, 1955.

[9] Kevin Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UC Berkeley, Computer Science Division, July 2002.

[10] A. Nerode and W. Kohn. Models for hybrid systems: Automata, topologies, controllability, observability. In R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel, editors, *Hybrid Systems*, number 736 in Lecture Notes on Computer Science, pages 317–356. Springer-Verlag, 1993.

[11] James L. Peterson. *Petri net theory and the modeling of systems*. Prentice -Hall, 1981.

[12] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, New York, 1994.

[13] Robert St-Aubin. *Probabilistic Constraint Nets: A Framework for the Modeling and Verification of Probabilistic Hybrid Systems*. PhD thesis, University of British Columbia, Department of Computer Science, www.cs.ubc.ca/spider/staubin/Papers/StAubin.pdf, June 2005.

[14] Robert St-Aubin and Alan K. Mackworth. Constraint-based approach to modeling and verification of probabilistic hybrid systems. Technical Report TR-2004-05, University of British Columbia, www.cs.ubc.ca/spider/staubin/Papers/TR-04-05.pdf, April 2004.

[15] Robert St-Aubin and Alan K. Mackworth. Modeling uncertain dynamical systems and solving behavioural constraints using probabilistic constraint nets. In *ECAI: Workshop on Modeling and Solving Problems with Constraints*, pages 70–85, Valencia, Spain, August 2004.

[16] A. Tarski. A lattice theoretical fixpoint theorem and its applications. *Pacific J. of Math*, 5:285–309, 1955.

[17] Ying Zhang and Alan K. Mackworth. Constraint nets: a semantic model for hybrid systems. *Theoretical Computer Science*, 138(1):211–239, 1995.

## 8 Appendix

## A Proofs of Theorems

Before presenting the proofs of the two fixpoint theorems, we need the following result.

**Proposition A** Any continuous (or pathwise continuous) function is monotonic, i.e., if $f : \Omega \times A \rightarrow A'$ ($f_\omega : A \rightarrow A'$) is continuous (pathwise continuous), then $(\omega_1, a_1) \leq_{\Omega\times A} (\omega_2, a_2)$ ($a_1 \leq_A a_2$) implies $f(\omega_1, a_1) \leq_{A'} f(\omega_2, a_2)$ ($f_\omega(a_1) \leq_{A'} f_\omega(a_2)$).

**Proof:** We prove this result for pathwise continuous functions. The result extends easily to continuous functions. Suppose $f_\omega(a_1) \not\leq_{A'} f_\omega(a_2)$, then according to the definition of partial order topology, there is an open set $S \subseteq A'$ including $f_\omega(a_1)$ but not $f_\omega(a_2)$. Therefore, $f_\omega^{-1}(S) \subseteq A$ is an open set including $a_1$ but not $a_2$. So $a_1 \not\leq_A a_2$. ■

Now we are ready to present the proofs for the fixpoint theorems introduced in Section 4.1.

**Theorem 4.1** Let $A$ be a *cpo* and assume that either $A$ is also a *total order* or that the set of distributions over $A$ is a *cpo* and the function over distributions is continuous. Then, every continuous function $f : \Omega \times A \rightarrow A$ or pathwise continuous function $f_\omega : A \rightarrow A$ (for a fixed $\omega \in \Omega$) has a least fixpoint in distribution.

**Proof:**

**When $A$ is a *total order*.**

To prove this results, we will use the classic *Tarski's fixpoint theorem* (Lattice-theoretical fixpoint theorem) [16]. Let us first introduce the theorem and then show how to use the result to prove our Fixpoint Theorem.

**Theorem A.1 (Tarski's Fixpoint Theorem)** *Let*

1. *$\mathscr{U} = \langle A, \leq\rangle$ be a complete lattice,*
2. *$f$ be a monotonically increasing function on $A$ to $A$,*
3. *$P$ be the set of all fixpoints of $f$.*

*Then the set $P$ is not empty and the system $\langle P, \leq\rangle$ is a* complete lattice.

**Proof:** (Theorem A.1) For the proof of this well-known result the reader is referred to the original work by Tarski [16]. ■

In order to be able to use Tarski's results, we need to show that the set of distributions and its partial order define a complete lattice. Moreover, we also need to show that the function $f$ on the set of distribution is monotonically increasing.

First, denote the set of all distributions on $A$ by $\mathscr{D}$. We formally define a partial order on the set of distributions $\mathscr{D}$. The binary relation $\leq_D$ on $\mathscr{D}$ is defined as follow. Let $F_{X_1}$ and $F_{X_2}$ be distributions of two random variables, namely $X_1$ and $X_2$. We write $F_{X_1} \leq_D F_{X_2}$, if $\forall a \in A$, $Pr(X_1 \leq a) \geq Pr(X_2 \leq a)$. It is easy to show that $\leq_D$ induces a partial order on $\mathscr{D}$.

Second, we need to show that for any two distributions $F_1, F_2 \in \mathscr{D}$, there exists a least upper bound and a greatest lower bound. To prove this, let us look at the *cumulative distribution function* (cdf) of each distribution. Here we reproduce Theorem 1.5.1 of [1] and refer to this reference for the proof.

**Theorem A.2 (Theorem 1.5.1 of Casela and Berger)** *The function $F(x)$ is a cdf if and only if the following three conditions hold:*

1. *$lim_{x\rightarrow-\infty}F(x) = 0$ and $lim_{x\rightarrow\infty}F(x) = 1$.*
2. *$F(x)$ is a nondecreasing function of $x$.*
3. *$F(x)$ is right-continuous. That is, for every number $x_0$, $lim_{x\downarrow x_0}F(x) = F(x_0)$.*

**Proof:** (Theorem A.2) See p. 30 of §1.5 from [1].

Since $A$ is a total order, for each $F \in \mathscr{D}$, we have a well defined cdf. Hence every $F \in \mathscr{D}$ possess the three properties of a formal cdf. Based on these properties, it is easy to show that the upper envelope of any set of cdf is a least upper bound (LUB) while the lower envelope of the cdfs is the greatest lower bound (GLB). Moreover, both the LUB and GLB can be showed to be cumulative distribution functions since they are nondecreasing, right-continuous and converge to 0 and 1 as $x \downarrow -\infty$ and $x \uparrow \infty$ respectively. This demonstrate that we have a complete lattice.

Now let us show that $f$ applied recursively generates a sequence of monotonically increasing distributions. First assume, without loss of generality, that $f$ is Markovian. Moreover, let us assume that at each transition, the events $\{\omega_1, \cdots, \omega_n\}$ are independent and chosen from the sample space $\Omega$. Order the events $\{\omega_1, \cdots, \omega_n\}$ such that $f_{\omega_1}(a) \leq_A f_{\omega_2}(a) \leq_A \cdots \leq_A f_{\omega_n}(a)$ for any $a \in A$. Let $\perp_A$ denote the least element of $A$ and let $F_X$ denote the distribution of the random variable $X$.

Now we want to prove that $F_{f^n(\perp_A)} \leq_D F_{f^{n+1}(\perp_A)}$.

*Proof by Induction on $n$:*

- For $n = 0$: We have that $F_{f^0(\perp_A)} =\perp_A$
- For $n = 1$: From Proposition A, since $f_\omega$ is continuous, we have that $f_\omega$ is also monotonic, $\forall\omega \in \Omega$. Hence we have $f_\omega(\perp_A) \geq_A \perp_A, \forall\omega \in \Omega$. Therefore, it is trivial to prove that $F_{f^0(\perp_A)} = F_{\perp_A} \leq_D F_{f^1(\perp_A)}$.
- Induction Hypothesis: Assume that for an arbitrary chosen $n \in \mathbb{N}$, $F_{f^n(\perp_A)}$ exists and is well-defined. We now need to show that $F_{f^n(\perp_a)} \leq_D F_{f^{n+1}(\perp_A)}$.

  Let $M_1 = max\{f^n(\perp_A)\} = \underbrace{f_{\omega_n} \circ \cdots \circ f_{\omega_n}}_{\text{n times}}(\perp_A)$. Based on this definition, we have $P(f^n(\perp_A) \leq M_1) = 1$.

  It is easy to show that $\underbrace{f_{\omega_n} \circ \cdots \circ f_{\omega_n}}_{\text{n times}} \circ f_{\omega_i} \geq M_1$ since $f_{\omega_n} \circ \cdots \circ f_{\omega_n}$ is monotonic.

Hence, we get the following result:

$$\begin{aligned} P(f^{n+1}(\bot_A) \leq M_1) & \leq 1 - \sum_{i=1}^{n} P(\omega_n \cdots \omega_n \omega_i) \\ & \leq 1 - \underbrace{P(\omega_n \cdots \omega_n \omega_i)}_{\text{finite and } >0} \\ & \text{since } \sum_{i=1}^{n} P(\omega_i) = 1 \\ & \leq 1 - P^n(\omega_n) \\ & < 1. \end{aligned} \tag{1}$$

Let $M_2 = max\{\{f^n(\bot_A)\} - M_1\}$ be the second highest value after $n$ iterations.

Say that $M_2$ arose from $f_{\omega^*}(\bot_A)$ where $\omega^* \in \omega_{i_1}\omega_{i_2}\cdots\omega_{i_n}$ with $\omega_i \in \Omega$. Then,

$$\begin{aligned} P(f^n(\bot_A) \leq M_2) & \leq 1 - P(\omega_n \cdots \omega_n) \\ & = 1 - P^n(\omega_n) \end{aligned} \tag{2}$$

Based on the same reasoning as above, we know that $f_{\omega^*} \circ f_{\omega_i}(\bot_A) \geq f_{\omega^*}(\bot_A)$. Therefore, we have

$$\begin{aligned} P(f^{n+1}(\bot_A) \leq M_2) & \leq 1 - \underbrace{P^n(\omega_n)}_{>0} - \underbrace{P(\omega^*)}_{>0} \\ & < 1 - P^n(\omega_n) \end{aligned} \tag{3}$$

By applying this reasoning until $M_n = min\{f^n(\bot_A)\} = \underbrace{f_{\omega_1} \circ \cdots \circ f_{\omega_1}}_{\text{n times}}(\bot_A)$, we get $P(f^n(\bot_A) \leq M_n) = P^n(\omega_1)$.

We know that $\underbrace{f_{\omega_1} \circ \cdots \circ f_{\omega_1}}_{\text{n times}} \circ f_{\omega_i}(\bot_A) \geq \underbrace{f_{\omega_1} \circ \cdots \circ f_{\omega_1}}_{\text{n times}}(\bot_A) = M_n$, which means that $P(f^{n+1}(\bot_A) \leq M_n) = 0$. We have proven that $F_{f^n(\bot_A)} \leq F_{f^{n+1}(\bot_A)}$ for any value of $n \in \mathbb{N}$. Hence, the transformational process on the distributions is a monotonically increasing one.

By applying Tarski's theorem, we have that the set of fixpoints of the distributions is non-empty and is a complete lattice. Therefore, there exist a least fixpoint in distribution and it concludes the proof under the assumption that $A$ is a total order. ■

**When the set of distributions over $A$, $\mathscr{D}$, is a *cpo* and the function over $\mathscr{D}$ is continuous.**

To prove this result, we can simply claim the following fixpoint theorem on cpos:

**Theorem A.3** *Let $\langle A, \leq, \bot_A\rangle$ be a cpo with least element $\bot_A$. Let $f : \langle A, \leq, \bot_A\rangle \rightarrow \langle A, \leq, \bot_A\rangle$ be a continuous function and let $\mu.f$ be the least upper bound of the chain $\{f^n(\bot_A) | n \in \mathbb{N}\}$. Then $\mu.f$ is the* least fixpoint *of $f$.*

**Proof:** (Theorem A.3) The proof can be found in any elementary algebraic theory textbooks such as [3].

This concludes the proof of the Fixpoint Theorem under the assumption that the set of distributions over $A$, $\mathscr{D}$, is a *cpo* and the function over $\mathscr{D}$ is continuous since we have satisfied all the necessary assumptions of the fixpoint theorem on a *cpo*. ■

**Theorem 4.2** Let $A$ and $A'$ be two *cpo*s and assume that either $A, A'$ are also *total orders* or that the set of distributions over $A'$ is a *cpo* and the function over distributions is continuous. If $f : \Omega \times A \times A' \rightarrow A'$ is a continuous function, then there exists a unique continuous function $\mu.f : \Omega \times A \rightarrow A'$, such that for all $a \in A$, the distribution of $(\mu.f)(a)$ is the least fixpoint in distribution of $\lambda\omega, x.f_\omega(a, x)$.

**Proof:** Let $F^0(a) = f(\omega, a, \bot_{A'})$ and $F^{k+1}(a) = f(\omega, a, F^k(a))$. Since $f$ is continuous, it is continuous w.r.t. the third argument. Moreover, a continuous function in any partial order is also monotonic. Therefore, for every $a$,

$$F^0(a) \leq_{A'} F^1(a) \leq_{A'} F^2(a) \cdots \leq_{A'} F^k(a) \leq \ldots.$$

The proof of the existence of the least fixpoint $\mu.f$ is left to the reader as it is very similar to the proof of Theorem 4.1.

Next, we prove that $\mu.f$ is continuous.

Clearly for every $k$, $F^k$ is continuous since $f$ is continuous and continuity is closed under functional composition. Therefore, for any directed subset $D$ of $A$,

$$\begin{aligned} \mu.f(\bigvee_A D) & = \bigvee_{A'}\{F^k(\bigvee_A D) | k \geq 0\} \\ & = \bigvee_{A'}\{\bigvee_{A'}\{F^k(D)\} | k \geq 0\} \\ & = \bigvee_{A'}\{\bigvee_{A'}\{F^k(a) | k \geq 0\} | a \in D\} \\ & = \bigvee_{A'} \mu.f(D). \end{aligned}$$

■