Refinement for Probabilistic Systems with Nondeterminism

Steve Reeves
Department of Computer Science
University of Waikato
Hamilton
New Zealand
stever@cs.waikato.ac.nz

David Streader
Department of Computer Science
University of Waikato
Hamilton
New Zealand
dstr@cs.waikato.ac.nz

Before we combine actions and probabilities two very obvious questions should be asked. Firstly, what does "the probability of an action" mean? Secondly, how does probability interact with nondeterminism? Neither question has a single universally agreed upon answer but by considering these questions at the outset we build a novel and hopefully intuitive probabilistic event-based formalism.

In previous work we have characterised refinement via the notion of testing. Basically, if one system passes all the tests that another system passes (and maybe more) we say the first system is a refinement of the second. This is, in our view, an important way of characterising refinement, via the question "what sort of refinement should I be using?"

We use testing in this paper as the basis for our refinement. We develop tests for probabilistic systems by analogy with the tests developed for non-probabilistic systems. We make sure that our probabilistic tests, when performed on non-probabilistic automata, give us refinement relations which agree with for those non-probabilistic automata. We formalise this property as a vertical refinement.

1 Introduction

Event-based models are frequently based on finite automata (FA, also called labelled transition systems) and probabilistic event-based systems are frequently based on FA where the transitions are also labelled by a probability as well as by an action. Before we combine events and probabilities two very obvious questions then arise. Firstly, what does "the probability of an event" mean, or what does it mean for an event to "behave in a probabilistic fashion"? Secondly, how does probability interact with nondeterminism? Neither question has a single universally agreed upon answer but by considering these questions at the outset we build a novel and hopefully intuitive probabilistic event-based formalism.

Throughout we will be motivated by a wish to, in the end, develop a notion of refinement for probabilistic systems. In fact, refinement will be the starting point of our story here as well as the desired end point.

In previous work we have characterised refinement via the notion of testing. Basically, if one system passes all the tests that another system passes (and maybe more) we say the first system is a refinement of the second. This is, in our view, an important way of characterising refinement since the question "what sort of refinement should I be using?" can be answered by saying "you should be using the sort of refinement that is characterised by the sort of tests which characterise the contexts within which your system will find itself, i.e. choose your refinement by looking at what contexts your systems will be used in."

Because this seems such a natural and useful answer, we use testing again in this paper as the basis for our refinement. We develop tests for probabilistic systems by analogy with the tests developed for non-probabilistic systems, all the while hoping to make sure that our probabilistic tests, when performed on non-probabilistic automata (and just noting whether a probability distribution is empty or not), give

J. Derrick , E.A. Boiten, S. Reeves (Eds.):
Refinement Workshop 2011.
EPTCS 55, 2011, pp. 84–100, doi:10.4204/EPTCS.55.6

us refinement relations which agree with for those non-probabilistic automata: this gives us confidence that our new notions make sense. We formalise this property in Section 7.

The real test (!) in all this comes when we consider probabilistic automata which also contain nondeterminism. Again, we are guided by the wish that our probabilistic tests, when used on nondeterministic, non-probabilistic automata, give us a refinement ordering which agrees with that originally given for those automata when probability was not considered. We also find that the algebraic properties that characterise the non-probabilistic case carry over into our new domain.

We formalise a notion of refinement based upon probabilistic tests and then try to (re-)capture what nondeterminism means in this probabilistic setting.

We will first introduce transition systems as a semantic foundation for non-probabilistic automata and recap previous work on using testing to define refinement for such systems.

It will turn out that part of the key to doing this for probabilistic systems is to be clear about two different philosophical bases for probability, so we next review those. Another part of the key to this work will be a consideration of how nondeterminism is characterised, so we will go on to discuss that subsequently. This will finally suggest how we might adapt transition systems to allow consideration of probability, and we finally show how this adaptation can be used to also allow a treatment of nondeterministic probabilistic systems, all the while retaining our testing-based notion of refinement.

We also show (via a selection) that expected properties hold for our refinement.

2 Transition systems

Definition 1 *Finite Automata (FA). Let Act be a set of actions and let Act^τ be the same set along with the special action τ, which represents actions interacting to form events. Let N_A be a finite set of nodes.*

The finite automaton A is given by the triple $(N_\mathsf{A}, S_\mathsf{A}, T_\mathsf{A})$ where

1. *$S_\mathsf{A} \subseteq N_\mathsf{A}$ is a set of start nodes*
2. *$T_\mathsf{A} \subseteq \{(n,\mathsf{a},m) | n,m \in N_\mathsf{A} \wedge \mathsf{a} \in Act^\tau\}$ shows the effect of each action.*

We write $x \xrightarrow{\mathsf{a}}_\mathsf{A} y$ for $(x,\mathsf{a},y) \in T_\mathsf{A}$ and $x \xrightarrow{\mathsf{a}} y$ where A is obvious from context. We write $n \xrightarrow{\mathsf{a}}$ for $\exists m.(n,\mathsf{a},m) \in T_\mathsf{A}$, and $m \xrightarrow{\rho} n$ for

$$\exists m_1 \ldots m_i . m \xrightarrow{\rho_1} m_1, m_1 \xrightarrow{\rho_2} m_2, \ldots m_i \xrightarrow{\rho_i} n$$

and $m \xrightarrow{\rho}$ for

$$\exists m_1 \ldots m_i, n . m \xrightarrow{\rho_1} m_1, m_1 \xrightarrow{\rho_2} m_2, \ldots m_i \xrightarrow{\rho_i} n$$

when $\rho = (\rho_1, ..., \rho_i)$, a finite sequence of actions.

We write $n \Longrightarrow m$ for $n \xrightarrow{\tau^*} m$, $n \xRightarrow{\mathsf{a}} m$ for $\exists j,k . n \Longrightarrow j \wedge j \xrightarrow{\mathsf{a}} k \wedge k \Longrightarrow m$ and $n \xRightarrow{\mathsf{a}}$ for $\exists j,k,m . n \Longrightarrow j \wedge j \xrightarrow{\mathsf{a}} k \wedge k \Longrightarrow m$.

$m \xRightarrow{\rho}$ and $m \xRightarrow{\rho} n$ are defined similarly to the cases for $\longrightarrow$.

Where ρ is a sequence of actions over Act^τ we write ρ_0 for ρ with the τs removed.

The traces are $Tr(\mathsf{A}) \stackrel{\text{def}}{=} \{\rho \mid s \in S_\mathsf{A} \wedge s \xRightarrow{\rho}\}$.

The complete traces[1] are $Tr^c(\mathsf{A}) \stackrel{\text{def}}{=} \{\rho \mid (s \in S_\mathsf{A} \wedge s \xRightarrow{\rho} n \wedge \pi(n) = \emptyset)$ where $\pi(n) \stackrel{\text{def}}{=} \{m \mid n \xrightarrow{\times}_\mathsf{A} m\}$.

[1] We deal with only acyclic automata and so we do not need to deal with infinite traces, though all the work of this paper can be extended to infinite traces and cyclic automata in the standard way [1].

We wish to model, using our automata, components that, like CSP processes, can immediately be nondeterministic. But, unlike CSP, we wish hiding (abstraction) to distribute through choice (so τs are used only for unobservable actions or for events, and not pressed into service to encode nondeterministic choice between starting states). There is a subtle difference between how external choice in CSP and choice in CCS behave with processes containing initial τ actions. This has been explained either by regarding the choice operators as being different, see [2] "The unique choice operator of CCS, denoted by +, is a mixture between external and internal choices" or by viewing CSP's use of τ actions to model a nondetermined start state as different to CCS's use of τ actions [3]. By allowing automata to have a set of start states we both avoid having to distinguish external choice and CCS choice and allow hiding to distribute through choice [3].

Also, choice can be defined ([4, 5]) between FAs with one start state each by gluing the two start states together to make a new single start state. Here, due to our generalisation, we glue together two sets of start states.

Let $S = \{s_1, s_2, \ldots, s_n\}$ and $S' = \{s'_1, s'_2, \ldots, s'_m\}$ be two sets of starting states and then define $\{S/S \times S'\}$ to be the n substitutions $\{s_i \in S | s_i / \{(s_i, s'_1), \ldots, (s_i, s'_m)\}\}$ and define $\{S'/S \times S'\}$ to be the m substitutions $\{s'_j \in S' | s'_j / \{(s_1, s'_j), \ldots, (s_n, s'_j)\}\}$.

We define $\{SS'/S \times S'\}$ to be the $n+m$ simultaneous substitutions $\{S/S \times S'\} \cup \{S'/S \times S'\}$. The first n substitutions replace each element of $\{s_1, s_2, \ldots, s_n\}$ with a set of m nodes and the last m substitutions simultaneously replace each element of $\{s'_1, s'_2, \ldots, s'_m\}$ with a set of n nodes. Consequently $\{S_\mathsf{A} S_\mathsf{B}/S_\mathsf{A} \times S_\mathsf{B}\}$ will identify the two sets of nodes S_A and S_B as $S_\mathsf{A}\{S_\mathsf{A} S_\mathsf{B}/S_\mathsf{A} \times S_\mathsf{B}\}$ and $S_\mathsf{B}\{S_\mathsf{A} S_\mathsf{B}/S_\mathsf{A} \times S_\mathsf{B}\}$ are both the $n \times m$ set of nodes $S_\mathsf{A} \times S_\mathsf{B}$.

Since single states may now become sets of states under the substitution, we also have to define what it means to have sets of nodes in a transition:

$$T \xrightarrow{x} T' \stackrel{\text{def}}{=} \{t \xrightarrow{x} t' | t \in T, t' \in T'\}$$

Definition 2 *Process operators. Let* A *be* $(N_\mathsf{A}, S_\mathsf{A}, T_\mathsf{A})$ *and let* B *be* $(N_\mathsf{B}, S_\mathsf{B}, T_\mathsf{B})$.
Action Prefixing $\mathsf{a}.\mathsf{B} \stackrel{\text{def}}{=} (\{s\} \cup N_\mathsf{B}, \{s\}, \{s \xrightarrow{\mathsf{a}} x | x \in S_\mathsf{B}\} \cup T_\mathsf{B})$ *where s is a new state.*
Internal choice $\mathsf{A} \sqcap \mathsf{B} \stackrel{\text{def}}{=} (N_\mathsf{A} \cup N_\mathsf{B}, S_\mathsf{A} \cup S_\mathsf{B}, T_\mathsf{A} \cup T_\mathsf{B})$
External choice is, informally, internal choice where start states are combined according to the substitutions above. Let $S_{\mathsf{A}\Box\mathsf{B}}$ *be* $\bigcup((S_\mathsf{A} \cup S_\mathsf{B})\{S_\mathsf{A} S_\mathsf{B}/S_\mathsf{A} \times S_\mathsf{B}\})$*, i.e. we combine start states as above. Then,*
External choice $\mathsf{A}\Box\mathsf{B} \stackrel{\text{def}}{=} ((N_\mathsf{A} \cup N_\mathsf{B}) \setminus (S_\mathsf{A} \cup S_\mathsf{B}) \cup S_{\mathsf{A}\Box\mathsf{B}}, S_{\mathsf{A}\Box\mathsf{B}}, (T_\mathsf{A} \cup T_\mathsf{B})\{S_\mathsf{A} S_\mathsf{B}/S_\mathsf{A} \times S_\mathsf{B}\})$
Parallel composition: $\mathsf{A} \parallel_P \mathsf{B} \stackrel{\text{def}}{=} (N_{\mathsf{A}\parallel_P\mathsf{B}}, S_{\mathsf{A}\parallel_P\mathsf{B}}, T_{\mathsf{A}\parallel_P\mathsf{B}})$ *where* $P \subseteq N_\mathsf{A} \cap N_\mathsf{B}$, $N_{\mathsf{A}\parallel_P\mathsf{B}} = N_\mathsf{A} \times N_\mathsf{B}$, $S_{\mathsf{A}\parallel_P\mathsf{B}} = S_\mathsf{A} \times S_\mathsf{B}$ *and* $T_{\mathsf{A}\parallel_P\mathsf{B}}$ *is defined by:*

$$\frac{n \xrightarrow{x}_\mathsf{A} l, m \xrightarrow{x}_\mathsf{B} k, x \in P}{(n,m) \xrightarrow{\tau}_{\mathsf{A}\parallel_P\mathsf{B}} (l,k)}$$

$$\frac{n \xrightarrow{x}_\mathsf{A} l, (x \notin P \wedge m \in N_\mathsf{B})}{(n,m) \xrightarrow{x}_{\mathsf{A}\parallel_P\mathsf{B}} (l,m)} \qquad \frac{n \xrightarrow{x}_\mathsf{B} l, (x \notin P \wedge m \in N_\mathsf{A})}{(m,n) \xrightarrow{x}_{\mathsf{A}\parallel_P\mathsf{B}} (m,l)}$$

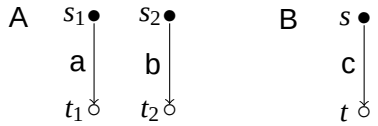
Example 1 *Let* A *be*

$$(\{s_1, s_2, t_1, t_2\}, \{s_1, s_2\}, \{s_1 \xrightarrow{\mathsf{a}}_\mathsf{A} t_1, s_2 \xrightarrow{\mathsf{b}}_\mathsf{A} t_2\})$$

and let B *be*

$$(\{s, s_2, t\}, \{s\}, \{s \xrightarrow{\mathsf{c}}_\mathsf{B} t\})$$

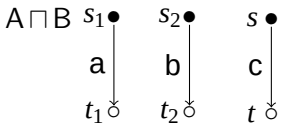
or, in diagram form,



Then $\mathsf{A} \sqcap \mathsf{B}$ *is*

$$(\{s_1, s_2, s, t_1, t_2, t\}, \{s_1, s_2, s\}, \{s_1 \xrightarrow{\mathsf{a}}_{\mathsf{A}\sqcap\mathsf{B}} t_1, s_2 \xrightarrow{\mathsf{b}}_{\mathsf{A}\sqcap\mathsf{B}} t_2, s \xrightarrow{\mathsf{c}}_{\mathsf{A}\sqcap\mathsf{B}} t\})$$

or, as a diagram,



Given that $S_{\mathsf{A}\Box\mathsf{B}}$ *is*

$$\bigcup \{s_1, s_2, s\}\{s_1/\{(s_1, s)\}, s_2/\{(s_2, s)\}, s/\{(s_1, s), (s_2, s)\}\} = \{(s_1, s), (s_2, s)\}$$

then $\mathsf{A}\Box\mathsf{B}$ *is*

$$(\{t_2, t_3, (s_1, s), (s_2, s)\}, \{(s_1, s), (s_2, s)\},$$

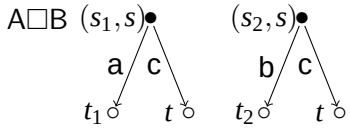
$$\{(s_1, s) \xrightarrow{\mathsf{a}}_{\mathsf{A}\Box\mathsf{B}} t_1, (s_2, s) \xrightarrow{\mathsf{b}}_{\mathsf{A}\Box\mathsf{B}} t_2, \{(s_1, s), (s_2, s)\} \xrightarrow{\mathsf{c}}_{\mathsf{A}\Box\mathsf{B}} t\})$$

which is

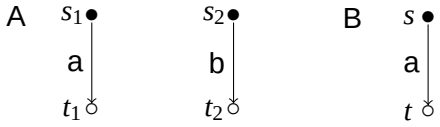
$$(\{t_2, t, (s_1, s), (s_2, s)\}, \{(s_1, s), (s_2, s)\},$$

$$\{(s_1, s) \xrightarrow{\mathsf{a}}_{\mathsf{A}\Box\mathsf{B}} t_1, (s_2, s) \xrightarrow{\mathsf{b}}_{\mathsf{A}\Box\mathsf{B}} t_2, (s_1, s) \xrightarrow{\mathsf{c}}_{\mathsf{A}\Box\mathsf{B}} t\}, (s_2, s) \xrightarrow{\mathsf{c}}_{\mathsf{A}\Box\mathsf{B}} t\})$$

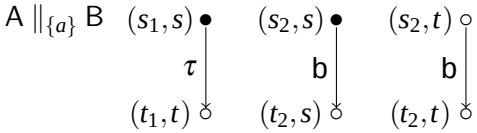
and as a diagram



Finally, $\mathsf{A} \parallel_{\{a\}} \mathsf{B}$ *with (note that* B*'s action is now* a*)*



is



□

3 Testing semantics

The definitions in this section are taken from [6] where they have been applied to both state-based and event-based models.

One of our tests, of a process E, taken from a set of processes $\mathbb{E}$, consists of placing E in some context X taken from a set of possible contexts Ξ. E in context X is written $[\mathsf{E}]_{\mathsf{X}}$. We then observe the resulting system. Each observation made is taken from a set of possible observations $\mathbb{O}$.

We turn first to our general definition of testing semantics for nondeterministic processes and contexts. In this setting a test may return (nondeterministically) one observation from a set of possible observations.

A specification is interpreted as a *contract* consisting of the *assumption* that the process will be placed only in one of the specified contexts Ξ and a *guarantee* that the observation of its behaviour will be one of the observations defined by the mapping $O : \mathbb{E} \to \Xi \to \wp\mathbb{O}$. The mapping O defines what can be observed for all processes in any of the assumed contexts. Hence for any fixed Ξ and O we have a definition of the semantics and the refinement of processes.

Definition 3 *Let Ξ be a set of contexts each of which the processes $\mathsf{A}, \mathsf{C} \in \mathbb{E}$ can communicate privately with, and let $O : \mathbb{E} \to \Xi \to \wp\mathbb{O}$ be a function which returns a set of observations, i.e. a subset of $\mathbb{O}$. Then, the relational semantics of a process A is a subset of $\Xi \times \mathbb{O}$.*

$$[\![\mathsf{A}]\!]_{\Xi,O} \stackrel{\text{def}}{=} \{(x,o) | x \in \Xi \wedge o \in O([\mathsf{A}]_x)\}$$

and refinement is given by

$$\mathsf{A} \sqsubseteq_{\Xi,O} \mathsf{C} \stackrel{\text{def}}{=} [\![\mathsf{C}]\!]_{\Xi,O} \subseteq [\![\mathsf{A}]\!]_{\Xi,O}$$

and equality is

$$\mathsf{A} =_{\Xi,O} \mathsf{C} \stackrel{\text{def}}{=} [\![\mathsf{C}]\!]_{\Xi,O} = [\![\mathsf{A}]\!]_{\Xi,O}$$

□

Given a rich enough class of tests the use of nondeterministic tests is redundant, as what can be observed using a nondeterministic test will be the union of what can be observed using a set of deterministic tests. Hence nondeterministic tests add no further information and will be ignored.

For all the processes considered in this paper, placing a process A in a context X, i.e. $[\mathsf{A}]_{\mathsf{X}}$, will mean executing process A in parallel with X, i.e. $\mathsf{A} \parallel_N \mathsf{X}$ (where N is some set of actions over which the context and process communicate, i.e. synchronize) and the observation function O is either the trace function Tr (if only safety properties are of interest) or (if liveness properties are of interest) the complete trace function Tr^c.

Definition 4 *Let Ξ_{FA} be FA and let $\sqsubseteq_{FA}$ be $\sqsubseteq_{\Xi_{FA},Tr^c}$.* □

Theorem 1 *Refinement distributes through parallel composition: Let $\mathsf{X}, \mathsf{Y}, \mathsf{P}, \mathsf{Q} \in FA$*

$$\frac{\mathsf{X} \sqsubseteq_{FA} \mathsf{Y}, \mathsf{P} \sqsubseteq_{FA} \mathsf{Q}}{\mathsf{X} \parallel_N \mathsf{P} \sqsubseteq_{FA} \mathsf{Y} \parallel_N \mathsf{Q}}$$

□

4 Probabilities—Two Interpretations

There are two (main) interpretations of probability, the *frequentist* and the *Bayesian*.

The frequentists' definition sees probability as the long-run expected frequency of occurrence. The probability of event A happening, where n is the number of times event A occurs in N opportunities, is $P(A) = n/N$.

The Bayesians' view of probability is related to degree of belief or state of knowledge. It is a measure of the plausibility of an event given incomplete knowledge. The Bayesian probabilist specifies some given or assumed prior probabilities, which are then used in the computation of other probabilities. That is to say, anything that is nondeterministic or unknown must either be assigned some probability or have its probability computed from other, more primitive, known probabilities. Bayesian statisticians have developed several "objective" methods for specifying prior probabilities.

The frequentists' view is based upon repeatedly performing the same test many times and, where the behaviour of the item under test is nondeterministic, aggregating the results of all the tests. Extending an event-based testing semantics to record not just the set of possible observations but the probability with which they occur is a simple uniform way to extend event-based testing semantics to event-based probabilistic testing semantics. This can be further generalised by representing both the process under test and the test process itself with probabilistic automata.

The Bayesian view fits well with Hoare's comment on nondeterminism [7, p81]:

> "*There is nothing mysterious about this kind of nondeterminism: it arises from a deliberate decision to ignore the factors which influence the selection*"

So, nondeterminism in a process is merely a case of not having analysed it enough to quantify it, i.e. attach to it some probabilities. Nondeterministic choice is probabilistic choice with unknown probabilities. Surprisingly, this is not how testing semantics have been defined in the literature.

As probabilities quantify (i.e. attach a number to, or make quantitative) nondeterministic behaviour, it is clearly crucial when modelling some real process to distinguish between the behaviour of the process being deterministic and the behaviour being nondeterministic. Similarly when the process is observed interacting in some context it is crucial to distinguish the nondeterminism of the process from the nondeterminism of the context.

Give a coin to a frequentist statistician and they experiment by flipping the coin a large number of times noting down the number of times they observe heads being uppermost and the number of times they observe tails. From this experiment they can compute the probability.

An important point to note is that, to the frequentist, probabilities define how likely it is that an action is executed, or equivalently how likely it is that the execution ends in a particular state. The probability of an event occurring when the event cannot be executed must be zero.

The Bayesian statistician, given a coin, knows that the only observations are heads and tails, and has no further information. The skill of the Bayesian statistician is to assign a prior probability based on understanding the world that agrees with the frequentist. It becomes very important when we try to add probabilities to event-based processes that we either follow the frequentist and perform experiments (tests) or follow the Bayesian statistician and think clearly about the behaviour in the world of what we are modelling.

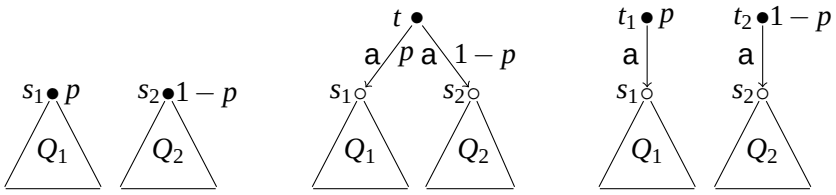


Figure 1: Probabilities on starting states

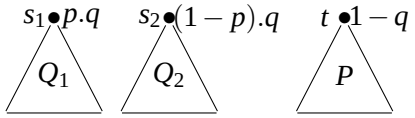


Figure 2: More general probabilistic combination

5 Probabilistic Finite Automata

5.1 Probability

We introduce probabilities on choice by attaching probabilities to the start states of a process. There are two things to notice here: as in the non-probabilistic case with FAs, we represent nondeterminism on the initial state of a process by allowing the process to start in one of a *set* of states; and we generalise this idea to represent the *probability* of starting in some state of a process by attaching probabilities to each of its start states so that we can see what the probability of each possible start state being actually chosen for some particular execution of the process.

The first of these points is inherited from work [8] which seeks to remove the need to use unobservable actions to also "encode" or represent nondeterminism in a process by assuming the process makes an unobserved transition to its "real" starting state (which may be one of many) from some single "dummy" formal starting state. (And, of course, this is just a case of using the usual "set of states" model uniformly for start states as well as all other states, which is something we are all familiar with from the "classic" algorithm that constructs a deterministic finite-state automaton from a nondeterministic one.) Such unobserved actions can then be used exclusively to denote (synchronisation between) events. This idea is, in the second point above, carried over into the probabilistic realm so that initial probabilistic choice is replaced by a probability distribution over the possible starting states.

So, if P is the process that starts with a choice between Q_1 and Q_2, which have (single, for this illustration) starting states s_1 and s_2 respectively, with probabilities p of starting in state s_1 and $1-p$ of starting in state s_2 then we might picture P as in the left of Figure 1. We might represent the picture by saying $S(P) = \{s_1 \mapsto p, s_2 \mapsto 1-p\}$, where S is a probability distribution function over start states of P.

Further, if we now form the process $a.P$ (i.e. the event a happens then the process P happens) then we might picture this as in the middle of Figure 1, and here notice how the probabilities have migrated to the occurrences of event a. This picture suggests that transitions now represent the effect of an action on an initial state moving the system, according to some probability distribution, to the next state, when it synchronizes with the same action in some other process, i.e. when the two actions combine to form an event which takes place with the indicated probability.

So in $a.P$, the action a has the potential to move us from state t to state s_1 with probability p and to s_2 with probability $1-p$ when synchronized to form an event which actually does take place

with the indicated probabilities. We formalise all this by saying that the transitions of $a.P$ include $\{t \xrightarrow{a} d \mid d(s_1) = p \wedge d(s_2) = 1-p\}$. An alternative picture might be as shown in the right of Figure 1, and here notice how the probabilities on the new start states for the new process $a.P$ have migrated from the old start states of P and we have $S(a.P) = \{t_1 \mapsto p, t_2 \mapsto 1-p\}$. This picture might be considered a useful, though perhaps more unusual, alternative way of thinking of our system in the previous picture.

Note that the original form of transitions as in FAs can be recovered by using the domain of the probability distribution function to tell us what the relevant post-states are.

As processes are combined together, the probabilities for the various component start states are combined to form the probabilities for the start states of the combination. As an example, see Figure 2, which shows what the resultant start-state probabilities are for $(Q_1 +_p Q_2) +_q P$, where s_1, s_2 and t are the start states for Q_1, Q_2 and P respectively.

5.2 Probability and nondeterminism

From statistics, the *law of large numbers* tells us that nondeterministic behaviour is the same as probabilistic behaviour where the probabilistic behaviour is unknown but can be found by repeating the right experiment a large number of times.

In process algebras τ actions indicate hidden, unobservable, uncontrollable actions or events (a special case being when two processes synchronize on some actions, which we consider to be private and uncontrollable). Remember Hoare's comment that we cited in Section 4. We have said above that we view this as agreeing with the Bayesian idea that probability indicates a lack of information.

As probabilities refer to frequencies of executable behaviour, i.e. the probability of an event occurring, they naturally occur on τ actions. The intuitive relationship between nondeterminism and probability is widely held. For example,

> *"nondeterminism represents possible choices that can be resolved in a wholly unpredictable way. With probabilistic constructs the resolution becomes predictable up to a point, in that it is quantified"* [9]

We can view this as saying that probabilistic processes contain more information than nondeterministic processes but less than deterministic processes. Consequently what can be observed in any single observation of a probabilistic process is the same as what can be observed of the underlying non-probabilistic process. But by aggregating the observations of a large number of executions we can compute a probability distribution or verify a previously computed probability distribution.

As τ events are built by composing two actions that are observable (via parallel composition, i.e. synchronization) it would be useful to find some way to compute the probability of the executable τ event from the prior "probabilities" of their observable parts. This we do below in Definition 8.

The addition of probabilities to *observable* actions where there is *no* nondeterminism has proven both hard to interpret and hard to formalise, especially when we want to ensure that the models have desirable properties. One reason, in our opinion, that this has turned out to be so hard to do is that the probabilities on the observable actions need, obviously, to define the behaviour of the processes not just in one context but in *all* contexts.[2]

[2]We go no further with this point in this paper, but note that, in the non-probabilistic setting, we have considered this previously in [10].

5.3 Nondeterminism

We represent nondeterminism not by a separate set of operators but by allowing probabilities to be denoted not just by real numbers in the range 0 to 1 but also by real-valued terms (in that range) that contain variables or parameters. This introduces the idea of a starting-state distribution which is not completely determined or which has undetermined aspects, and hence allows us to represent nondeterminism with the same machinery that we introduce for probabilities.

This idea is motivated by the Bayesian view that the more we know about a mechanism, the more certain we can be about the probabilities attached to its behaviour: to talk of nondeterministic behaviour is merely to admit having more or less incomplete information about how something behaves, and this incomplete information can be represented by having parameters in the terms which denote probabilities. This also accords with Hoare's view that nondeterminism arises from ignoring or hiding (or, we would go further and say, being ignorant of) some aspects of a process. Further analysis of the mechanism would uncover ("unhide") more of the mechanism. This view dissolves nondeterminism; there is no such thing really, since it is just arises from not knowing (for whatever reason) enough about the actual distribution of probabilities amongst actions that might be taken when a choice is presented or confronted.

5.4 Probabilistic testing semantics

For probabilistic tests all we need change is that the user records not just a set of observations but a probability distribution over a set of observations, hence $\mathbb{O} \stackrel{\text{def}}{=} Act^* \to \mathbb{R}$.

The relational semantics of process A when probability distributions are observed is a subset of $\Xi \times (Act^* \to \mathbb{R})$. If a process is experimented upon (frequentist perspective) and the results noted then what is observed will be a function $\Xi \to (Act^* \to \mathbb{R})$ and hence there is no nondeterminism and no possibility of refinement.

But approaching automata from the Bayesian perspective, if we can define the processes and tests as prior "probabilistic" automata then we might be able to use probabilistic parallel composition to compute the probabilistic relational semantics of the processes. From the Bayesian point of view, the probabilities on actions are prior probabilities that, until the action takes part in an event by being synchronized with another process along the same action, do not play any role. Obviously the probability of an unexecuted action is prior to the probability of an execution—in particular, not until we factor in the probability of the synchronizing action do we know (via their product) what the probability of the executed event (denoted by τ) will be. So, it is the Bayesian ideas that allow us to make sense of attaching probabilities to something that has not yet happened, and which will only be a part of what happens.

6 Formalising probabilistic automata

In this section we will formalise the discussion in Section 5.2 and see that automata that contain both probabilistic and nondeterministic choice are called *partially probabilistic* introduced as parameterised probabilistic finite automata (PPFA). Here we take what we see as the standard statistical approach and model nondeterministic choice as probabilistic choice with unknown probability. So our probabilities are no longer only real numbers but may also be real-valued terms (parameterised terms, hence the name) that may contain variables, the unknown probabilities. Automata where nondeterminism has been completely replaced by probabilistic choice are *deterministic* probabilistic finite automata (DPFA).

Definition 5 *Parameterised Probabilistic Finite Automata (PPFA). Let N_{A} be a finite set of nodes. The parameterised probabilistic finite automaton A is given by the triple $(N_{\mathsf{A}}, S_{\mathsf{A}}, T_{\mathsf{A}})$ where*

1. *S_{A} is a "starting distribution", i.e. a parameterised probability distribution such that $dom(S_{\mathsf{A}}) \subseteq N_{\mathsf{A}}$, where $dom(S_A)$ are the starting states of* A
2. *$T_{\mathsf{A}} \subseteq \{(n, \mathsf{a}, d) | n \in N_{\mathsf{A}} \wedge \mathsf{a} \in Act^{\tau} \wedge d \in D_{\mathsf{A}}\}$, such that for each $n \in N_{\mathsf{A}}$ and $a \in Act^{\tau}$ there exists no more than one element of T_{A} with first component n and second component* a*, and recall that nondeterminism is modelled by a parameter in the range of the probability distribution d. Finally, D_A is a set of probability distributions over states.*

Deterministic Probabilistic Finite Automata (DPFA) are PPFA with the restrictions that:

1. *The ranges of all probability distributions are sets of real values, not sets of possibly parameterised terms, i.e. the elements of the ranges contain no variables;*
2. *$(n, \mathsf{a}, d) \in T_{\mathsf{A}}$ implies $\mathsf{a} \in Act$.*

□

Let the variables $\mathbb{X}, \mathbb{Y}$ be taken from some set *Var* and $\overline{\mathbb{X}}$ be a list of variables and $\psi_{\overline{\mathbb{X}}}$ be an instantiation of the variables in the list taken from the set of all such instantiations $\Psi_{\overline{\mathbb{X}}}$. We will write $\mathsf{A}(\overline{\mathbb{X}})$ for a PPFA containing variables $\overline{\mathbb{X}}$, but where not needed the list of variables will be dropped and we will write A. We interpret the variables in $\mathsf{A}(\overline{\mathbb{X}})$ as being *globally bound* and take the usual α-congruence of terms and identify PPFA that differ only by the names of variables used. Similarly we assume α-renaming to prevent confusion and variable capture when composing PPFAs.

We write $x \xrightarrow{\mathsf{a}}_{\mathsf{A},p} y$ for $(x, \mathsf{a}, d) \in T_{\mathsf{A}} \wedge d(y) = p$ and $x \xrightarrow{\mathsf{a}}_{p} y$ where A is obvious from context. In addition when we want to talk about a "complete" transition, i.e. one that has its associated final state distribution, we write $x \xrightarrow{\mathsf{a}}_{\mathsf{A}} d$ for $(x, \mathsf{a}, d) \in T_{\mathsf{A}}$.

Definition 6 *The probability of the computation following a path, a sequence of transitions starting from a start state s, is the product of the probability of its component transitions and the probability of starting in the start state $S_{\mathsf{A}}(s)$. Let p be the path $s \xrightarrow{\rho_1}_{p_1} m_1, m_1 \xrightarrow{\rho_2}_{p_2} m_2, \ldots m_{n-1} \xrightarrow{\rho_n}_{p_n} m_n$. Then the probability that p is executed is*

$$d(p) \stackrel{\text{def}}{=} S_{\mathsf{A}}(s) \times p_1 \times p_2 \times \ldots p_n$$

and we say that the path p can be observed as trace $\rho = \rho_1, \rho_2, \ldots, \rho_n$.

The probability of observing a trace ρ is the sum of the all probabilities of the computation following any path that can be observed as trace ρ:

$$d(\rho) = \sum_{tr(p_i)=\rho} d(p_i)$$

where $tr(p) \stackrel{\text{def}}{=} \{\rho | p = s \xrightarrow{\rho_1}_{p_1} m_1, m_1 \xrightarrow{\rho_2}_{p_2} m_2, \ldots m_{n-1} \xrightarrow{\rho_n}_{p_n} m_n\}$.

Writing $S_{\mathsf{A}} \xrightarrow{\rho}_{p}$ informs us that p is the probability of seeing the trace ρ when starting in any of the start states in $dom(S_{\mathsf{A}})$ and following some appropriate path, i.e. $d(\rho) = p$. $S_{\mathsf{A}} \xrightarrow{\rho}_{p} n$ means that p is the probability of seeing the trace ρ when starting in any of the start states in $dom(S_{\mathsf{A}})$ and ending in state n.

Definition 7 *The probability distribution over complete traces is*

$$D^c(\mathsf{A}) \stackrel{\text{def}}{=} \{\rho \mapsto \sum_{q \in P} q \mid P = \{q \mid n \in N_{\mathsf{A}} \wedge \pi(n) = \emptyset \wedge S_{\mathsf{A}} \xrightarrow{\rho}_{q} n\}\}$$

Definition 8 *Process operators*
Action Prefixing $\mathsf{a}.\mathsf{B} \stackrel{\text{def}}{=} (\{s_a\} \cup N_\mathsf{B}, \{s_a \mapsto 1\}, \{s_a \xrightarrow{\mathsf{a}} S_\mathsf{B}\} \cup T_\mathsf{B})$ *where* s_a *is a new state*
Internal choice $\mathsf{A} \sqcap \mathsf{B} \stackrel{\text{def}}{=} (N_\mathsf{A} \cup N_\mathsf{B}, S_\mathsf{A} \sqcap S_\mathsf{B}, T_\mathsf{A} \cup T_\mathsf{B})$ *where*
$(S_\mathsf{A} \sqcap S_\mathsf{B})(n) = \mathbb{X} \times S_\mathsf{A}(n)$ *if* $n \in dom(S_\mathsf{A})$ *else* $(1-\mathbb{X}) \times S_\mathsf{B}(n)$ *if* $n \in dom(S_\mathsf{B})$*, where* $\mathbb{X}$ *is a fresh parameter, and note that now* $dom(S_\mathsf{A} \sqcap S_\mathsf{B}) = dom(S_\mathsf{A}) \cup dom(S_\mathsf{B})$.
Probabilistic choice $\mathsf{A} \oplus_p \mathsf{B} \stackrel{\text{def}}{=} (N_\mathsf{A} \cup N_\mathsf{B}, S_\mathsf{A} \oplus_p S_\mathsf{B}, T_\mathsf{A} \cup T_\mathsf{B})$ *where* $(S_\mathsf{A} \oplus_p S_\mathsf{B})(n) = p \times S_\mathsf{A}(n)$ *if* $n \in dom(S_\mathsf{A})$ *else* $(1-p) \times S_\mathsf{B}(n)$ *if* $n \in dom(S_\mathsf{B})$*, and note that now* $dom(S_\mathsf{A} \oplus_p S_\mathsf{B}) = dom(S_\mathsf{A}) \cup dom(S_\mathsf{B})$. *We note immediately from this that internal choice is probabilistic choice with unknown probability between the two choices.*
External choice $\mathsf{A} \Box \mathsf{B} \stackrel{\text{def}}{=} (N_\mathsf{A} \cup N_\mathsf{B} \setminus (dom(S_\mathsf{A}) \cup dom(S_\mathsf{B})) \cup dom(S_{\mathsf{A}\Box\mathsf{B}}), S_{\mathsf{A}\Box\mathsf{B}}, T_\mathsf{A} \cup T_\mathsf{B}\{\{S_\mathsf{A}S_\mathsf{B}/S_\mathsf{A} \times S_\mathsf{B}\}\})$ *where* $S_{\mathsf{A}\Box\mathsf{B}}(n_\mathsf{A}, n_\mathsf{B}) = S_\mathsf{A}(n_\mathsf{A}) \times S_\mathsf{B}(n_\mathsf{B})$ *and* $\{\{S_\mathsf{A}S_\mathsf{B}/S_\mathsf{A} \times S_\mathsf{B}\}\}$ *now, of course, uses the* domains *of the start state distributions in order to build the substitutions over start states.*
Parallel composition:

$$\mathsf{A} \parallel_P \mathsf{B} \stackrel{\text{def}}{=} (N_{\mathsf{A}\parallel_P\mathsf{B}}, S_{\mathsf{A}\parallel_P\mathsf{B}}, T_{\mathsf{A}\parallel_P\mathsf{B}})$$

$$N_{\mathsf{A}\parallel_P\mathsf{B}} = N_\mathsf{A} \times N_\mathsf{B}$$

$$S_{\mathsf{A}\parallel_P\mathsf{B}}(n_\mathsf{A}, n_\mathsf{B}) = S_\mathsf{A}(n_\mathsf{A}) \times S_\mathsf{B}(n_\mathsf{B}) \; if \; n_\mathsf{A} \in dom(S_\mathsf{A}) \wedge n_\mathsf{B} \in dom(S_\mathsf{B})$$

and $T_{\mathsf{A}\parallel_P\mathsf{B}}$ *is defined by:*

$$\frac{n \xrightarrow{\mathsf{x}}_\mathsf{A} d_\mathsf{A}, m \xrightarrow{\mathsf{x}}_\mathsf{B} d_\mathsf{B}, \mathsf{x} \in P}{(n,m) \xrightarrow{\tau}_{(\mathsf{A}\parallel_P\mathsf{B})} d_\mathsf{A} \times d_\mathsf{B}}$$

$$\frac{n \xrightarrow{\mathsf{x}}_\mathsf{A} d_\mathsf{A}, (\mathsf{x} \notin P \wedge m \in N_\mathsf{B})}{(n,m) \xrightarrow{\mathsf{x}}_{(\mathsf{A}\parallel_P\mathsf{B})} d_\mathsf{A} \times m} \qquad \frac{n \xrightarrow{\mathsf{x}}_\mathsf{B} d_\mathsf{B}, (\mathsf{x} \notin P \wedge m \in N_\mathsf{A})}{(m,n) \xrightarrow{\mathsf{x}}_{(\mathsf{A}\parallel_P\mathsf{B})} m \times d_\mathsf{B}}$$

where

$$d_\mathsf{A} \times d_\mathsf{B} \stackrel{\text{def}}{=} \{(x,y) \mapsto d_\mathsf{A}(x).d_\mathsf{B}(y) | n \xrightarrow{\mathsf{x}}_A x \wedge m \xrightarrow{\mathsf{x}}_B y\}$$

and

$$d_\mathsf{A} \times m \stackrel{\text{def}}{=} \{(x,m) \mapsto d_\mathsf{A}(x) | n \xrightarrow{\mathsf{x}}_\mathsf{A} x\}$$

and

$$m \times d_\mathsf{B} \stackrel{\text{def}}{=} \{(m,y) \mapsto d_\mathsf{B}(y) | n \xrightarrow{\mathsf{x}}_\mathsf{B} y\}$$

Example 2 *Consider the PPFAs given by the expressions* $a.(Q_1 +_p Q_2)$ *and* $a.Q_1 +_p a.Q_2$*. Then, assuming the start states, states and transitions of* Q_1 *and* Q_2 *are given by* s_1*,* s_2*,* N_1*,* N_2*,* T_1 *and* T_2 *respectively, we have*

$$a.Q_1 +_p a.Q_2 = (\{t_1, t_2\} \cup N_1 \cup N_2, \{t_1 \mapsto p, t_2 \mapsto 1-p\},$$
$$\{t_1 \xrightarrow{\mathsf{a}} d_1, t_2 \xrightarrow{\mathsf{a}} d_2 | d_1(s_1) = d_2(s_2) = 1\} \cup T_1 \cup T_2)$$

$$a.(Q_1 +_p Q_2) = (\{t\} \cup N_1 \cup N_2, \{t \mapsto 1\},$$
$$\{t \xrightarrow{\mathsf{a}} d | d(s_1) = p, d(s_2) = 1-p\} \cup T_1 \cup T_2)$$

In fact, these PPFAs are indistinguishable by testing, so they are equal (they "refine both ways") as far as our testing semantics goes. This result can be generalised so that probability distributions on transitions can always be "migrated" to the starting state distribution.

6.1 Testing of probabilistic processes

Recall from Section 3 that we said in the definition of our testing semantics for FA that we will use $[\mathsf{A}]_{\mathsf{X}} = \mathsf{A} \parallel_N \mathsf{X}$ and $O_{FA} = Tr^c$. For probabilistic FAs we need to use parallel composition from Definition 8 (as defined for DPFA and PPFA). The observation of a single execution of a DPFA is still a trace but what can be "observed" over many executions is no longer simply a set of traces but, if we also record the frequency of occurrence of the traces, a probability distribution over the set of traces hence $O_{DPFA} = D^c$. We treat PPFA similarly and let $\Xi_{PPFA} = PPFA$ and $O_{PPFA} = D^c$ except that now the observed probability distributions may be parameterised.

Definition 9 *The relational semantics of an entity* $\mathsf{A}(\overline{\mathbb{X}})$ *is (where* $\Psi_{\overline{\mathbb{X}}}$ *is the set of instantiations for the parameters in* $\overline{\mathbb{X}}$*)*

$$[\![\mathsf{A}(\overline{\mathbb{X}})]\!]_{\Xi_{PPFA},D^c} \stackrel{\text{def}}{=} \{(x,o).x \in \Xi_{PPFA} \wedge o \in \psi_{\overline{\mathbb{X}}}(D^c(([\mathsf{A}(\overline{\mathbb{X}})]_x))) \wedge \psi_{\overline{\mathbb{X}}} \in \Psi_{\overline{\mathbb{X}}}\}$$

$$\mathsf{A}(\overline{\mathbb{X}}) \sqsubseteq_{\Xi_{PPFA},D^c} \mathsf{C}(\overline{\mathbb{Y}}) \stackrel{\text{def}}{=} [\![\mathsf{C}(\overline{\mathbb{Y}})]\!]_{\Xi_{PPFA},D^c} \subseteq [\![\mathsf{A}(\overline{\mathbb{X}})]\!]_{\Xi_{PPFA},D^c}$$

$$\mathsf{A}(\overline{\mathbb{X}}) =_{PPFA} \mathsf{C}(\overline{\mathbb{Y}}) \stackrel{\text{def}}{=} [\![\mathsf{C}(\overline{\mathbb{Y}})]\!]_{\Xi_{PPFA},D^c} = [\![\mathsf{A}(\overline{\mathbb{X}})]\!]_{\Xi_{PPFA},D^c}$$

Note here that we have given the meaning of PPFAs as a relation from contexts (PPFAs) to probability distributions:

$$[\![\mathsf{A}(\overline{\mathbb{X}})]\!]_{\Xi_{PPFA},D^c} \subseteq \Xi_{PPFA} \times (Act^* \to Real)$$

by instantiating all the open distributions that might be observed to get plain probability distributions "with no unknowns".

Let $\sqsubseteq_{PPFA} \stackrel{\text{def}}{=} \sqsubseteq_{\Xi_{PPFA},D^c}$. That is, we write $\sqsubseteq_{PPFA}$ for this general definition of refinement. When $\sqsubseteq_{PPFA}$ relates two DPFA processes it is of little interest, i.e. there are no opportunities for refinement as there is no nondeterminism (though there are, perhaps, probabilities).

In Section 7 we will show refinement of PPFA is strongly related to refinement of an underlying FA.

6.2 Simple results from the definitions

Theorem 2 *Refinement distributes through parallel composition. Let* X, Y, P *and* Q *be arbitrary PPFAs and let* $N \subseteq Act$*. Then*

$$\frac{\mathsf{X} \sqsubseteq_{PPFA} \mathsf{Y}, \mathsf{P} \sqsubseteq_{PPFA} \mathsf{Q}}{\mathsf{X} \parallel_N \mathsf{P} \sqsubseteq_{PPFA} \mathsf{Y} \parallel_N \mathsf{Q}}$$

For an arbitrary PPFA $\mathsf{P}(\overline{\mathbb{Y}})$ we have the following theorems.

Theorem 3 $\sqcap$ *is idempotent.* $\mathsf{P}(\overline{\mathbb{Y}}) =_{PPFA} \mathsf{P}(\overline{\mathbb{Y}}) \sqcap \mathsf{P}(\overline{\mathbb{Y}})$

Proof: From Definition 8 it can be seen that the graph of $\mathsf{P}(\overline{\mathbb{Y}}) \sqcap \mathsf{P}(\overline{\mathbb{Y}})$ consists of two copies of the graph of $\mathsf{P}(\overline{\mathbb{Y}})$ which ever copy is selected the behaviour is exactly that of $\mathsf{P}(\overline{\mathbb{Y}})$. Hence he equality.

Theorem 4 $\oplus_p$ *is idempotent* $\mathsf{P}(\overline{\mathbb{Y}}) =_{PPFA} \mathsf{P}(\overline{\mathbb{Y}}) \oplus_p \mathsf{P}(\overline{\mathbb{Y}})$

Proof: Similar to Theorem 3.

7 Relating finite automata to parameterised probabilistic finite automata

We construct $[\![\_]\!]^{FA}_{PPFA}$, an embedding of FA into PPFA and a forgetful mapping from PPFA to FA, and then show that these mappings form a Galois connection between the refinement relations $\sqsubseteq_{PPFA}$ and $\sqsubseteq_{FA}$.

The embedding $[\![\_]\!]^{FA}_{PPFA}$ of FA in PPFA will map all nondeterministic choices in FA processes into probabilistic choice with unknown probabilities in the PPFA processes.

Definition 10 *Semantic mappings $[\![\_]\!]^{FA}_{PPFA}$ and vA^{FA}_{PPFA} between finite automata A and parameterised probabilistic finite automata Ap are defined so that:*

$$[\![(N_{\mathsf{A}}, S_{\mathsf{A}}, T_{\mathsf{A}})]\!]^{FA}_{PPFA} \stackrel{\text{def}}{=} (N_{\mathsf{Ap}}, S_{\mathsf{Ap}}, T_{\mathsf{Ap}})$$

where

$$N_{\mathsf{Ap}} \stackrel{\text{def}}{=} N_{\mathsf{A}}$$

and

$$S_{\mathsf{Ap}} \stackrel{\text{def}}{=} \{(s, \mathbb{X}) \mid s \in S_{\mathsf{A}} \wedge \mathbb{X} \; is \; fresh \wedge (\Sigma_{n \in dom(S_{\mathsf{Ap}})} S_{\mathsf{Ap}}(n)) = 1\}$$

$$T_{\mathsf{Ap}} = \{(n, \mathsf{a}, d) \mid d = \{m \mapsto v \mid n \xrightarrow{\mathsf{a}} m \wedge v \; is \; fresh\} \wedge (\Sigma_{m \in dom(d)} d(m)) = 1\}$$

The mapping vA^{FA}_{PPFA} from PPFA in to FA forgets all probability distributions:

$$vA^{FA}_{PPFA}(N_{\mathsf{Ap}}, s_{\mathsf{Ap}}, T_{\mathsf{Ap}}) = (N_{\mathsf{A}}, s_{\mathsf{A}}, T_{\mathsf{A}})$$

where

$$N_{\mathsf{A}} \stackrel{\text{def}}{=} N_{\mathsf{Ap}}$$

and

$$S_{\mathsf{A}} \stackrel{\text{def}}{=} dom(S_{\mathsf{Sp}})$$

and

$$T_{\mathsf{A}} = \{(n, \mathsf{a}, m) | n \xrightarrow{\mathsf{a}}_{\mathsf{Ap}} d \wedge m \in dom(d)\}$$

□

The pair of mappings $([\![\_]\!]^{FA}_{PPFA}, vA^{FA}_{PPFA})$ define a vertical refinement $\sqsubseteq^{FA}_{PPFA}$ as they are a Galois connection [10]. This is the content of Theorem 7, but first some preliminary results.

Lemma 1 *For any FAs X and Y*
$Tr^c(\mathsf{X}) \subseteq Tr^c(\mathsf{Y}) \Rightarrow D^c([\![\mathsf{X}]\!]^{FA}_{PPFA}) \subseteq D^c([\![\mathsf{Y}]\!]^{FA}_{PPFA})$

Proof (Sketch) The application of $[\![\_]\!]^{FA}_{PPFA}$ to a FA simply adds parameterised probabilities spanning any nondeterministic choice. The set of all possible observation traces is $Tr^c(\mathsf{X})$. This is also the set of all possible observation traces of $[\![\mathsf{X}]\!]^{FA}_{PPFA}$ but now what is "observed" is not one trace but any probability distribution over any subset of $O(\mathsf{X})$ (we need to use subset as when the probability of observing a trace is 0 it is no longer in the domain of the distribution).

Hence $d \in D^c([\![\mathsf{X}]\!]^{FA}_{PPFA}) \Leftrightarrow dom(d) \subseteq Tr^c(\mathsf{X})$. Consequently if $d \in D^c([\![\mathsf{X}]\!]^{FA}_{PPFA})$ then $dom(d) \subseteq Tr^c(\mathsf{X})$ and since $Tr^c(\mathsf{X}) \subseteq Tr^c(\mathsf{Y})$, from the assumption of the lemma, we further have $dom(d) \subseteq Tr^c(\mathsf{Y})$. Then $d \in D^c([\![\mathsf{Y}]\!]^{FA}_{PPFA})$ follows from the argument above with Y in place of X. •

Theorem 5 *Let* X *and* Y *be FAs, and let* $N \subseteq Act$*. Then,*

$$[\![\mathsf{X} \parallel_N \mathsf{Y}]\!]^{FA}_{PPFA} = [\![\mathsf{X}]\!]^{FA}_{PPFA} \parallel_N [\![\mathsf{Y}]\!]^{FA}_{PPFA}$$

Theorem 6 *Let* X *and* Y *be PPFAs, and let* $N \subseteq Act$*. Then,*

$$vA^{FA}_{PPFA}(\mathsf{X} \parallel_N \mathsf{Y}) = vA^{FA}_{PPFA}(\mathsf{X}) \parallel_N vA^{FA}_{PPFA}(\mathsf{Y})$$

Definition 11 *Deterministic automata.*

$Det_{FA} \stackrel{\text{def}}{=} \{\mathsf{P} | (n \xrightarrow{\mathsf{a}} k \wedge n \xrightarrow{\mathsf{a}} l \Rightarrow k = l) \wedge |S_{\mathsf{A}}| = 1\}$

$Det_{PPFA} \stackrel{\text{def}}{=} \{\mathsf{P} | (n \xrightarrow{\mathsf{a}}_p k \wedge n \xrightarrow{\mathsf{a}}_q l \Rightarrow k = l \wedge p = q = 1) \wedge |S_{\mathsf{A}}| = 1\}$

Lemma 2 *Results involving deterministic automata.*

1. *(a)* $\{\mathsf{X} \in Det_{FA} \mid [\![\mathsf{X}]\!]^{FA}_{PPFA}\} = Det_{PPFA}$ *and*
   (b) $\{\mathsf{Y} \in Det_{PPFA} \mid vA^{FA}_{PPFA}(\mathsf{Y})\} = Det_{FA}$
2. *Let* A *and* C *be FAs. Then* $\mathsf{A} \sqsubseteq_{FA} \mathsf{C} \Leftrightarrow \forall_{x \in Det_{FA}}. Tr^c([\mathsf{A}]_x) \supseteq Tr^c([\mathsf{C}]_x)$
3. *Let* A *and* C *be PPFAs. Then* $\mathsf{A} \sqsubseteq_{PPFA} \mathsf{C} \Leftrightarrow \forall_{x \in Det_{PPFA}}. D^c([\mathsf{A}]_x) \supseteq D^c([\mathsf{C}]_x)$

Proof (Sketch).

$1(a)$ and $1(b)$ follow from definitions.

Re 2: With non-probabilistic processes and tests, what can be observed when applying a nondeterministic test is the union of what can be observed when applying each element of the set of deterministic alternatives (where here we picture, as usual, a nondeterministic computation as a set of deterministic ones which covers all the possible choices) and hence:

$$\mathsf{A} \sqsubseteq_{FA} \mathsf{C} \Leftrightarrow \forall_{x \in Det_{FA}}. Tr^c([\mathsf{A}]_x) \supseteq Tr^c([\mathsf{C}]_x)$$

Re 3: With probabilistic processes and tests, what can be observed when applying a probabilistic test is the distribution, inferred from the test, of what can be observed when applying the deterministic components that the probabilistic choice spans. Hence a set of test processes for PPFA that is sufficient to establish refinement is the image after applying $[\![\_]\!]^{FA}_{PPFA}$ to a sufficient set of FA processes, i.e. since Det_{FA} is sufficient for FA then Det_{PPFA} is sufficient for PPFA, hence:

$$\mathsf{A} \sqsubseteq_{PPFA} \mathsf{C} \Leftrightarrow \forall_{x \in Det_{PPFA}}. D^c([\mathsf{A}]_x) \supseteq D^c([\mathsf{C}]_x)$$

Theorem 7

$$\forall \mathsf{X} \in FA, \mathsf{Y} \in PPFA. [\![\mathsf{X}]\!]^{FA}_{PPFA} \sqsubseteq_{PPFA} \mathsf{Y} \Leftrightarrow \mathsf{X} \sqsubseteq_{FA} vA^{FA}_{PPFA}(\mathsf{Y})$$

Proof: (Sketch)

It is a well-known result (e.g. [11]) that to prove a Galois connection it is sufficient to prove for arbitrary X

$$vA^{FA}_{PPFA}([\![\mathsf{X}]\!]^{FA}_{PPFA}) \sqsubseteq_{FA} id_{FA}\mathsf{X}$$

and for arbitrary Y

$$[\![vA^{FA}_{PPFA}(\mathsf{Y})]\!]^{FA}_{PPFA} \sqsubseteq_{PPFA} id_{PPFA}\mathsf{Y}$$

and in addition to prove both relations $[\![\_]\!]^{FA}_{PPFA}$ and vA^{FA}_{PPFA} are monotone.

We can see directly from the definitions that $[\![\_]\!]^{FA}_{PPFA}$ adds parameterised probabilities to any nondeterministic choice and vA^{FA}_{PPFA} forgets this addition hence, for arbitrary X :

$$vA^{FA}_{PPFA}([\![\mathsf{X}]\!]^{FA}_{PPFA}) =_{FA} id_{FA}\mathsf{X}$$

which gives our first inequality.

The effect of $[\![vA^{FA}_{PPFA}\mathsf{Y}]\!]^{FA}_{PPFA}$ is to first replace probabilistic choice with nondeterministic choice (by ignoring probabilities) and then reintroducing probabilities-with-parameters due to the nondeterminism and this can be refined, along with other possibilities, back into its original value, which gives our second inequality.

Re: show $[\![\_]\!]^{FA}_{PPFA}$ is monotone: $\mathsf{A} \sqsubseteq_{FA} \mathsf{C} \Rightarrow [\![\mathsf{A}]\!]^{FA}_{PPFA} \sqsubseteq_{PPFA} [\![\mathsf{C}]\!]^{FA}_{PPFA}$

From Definition 3 we have $\mathsf{A} \sqsubseteq_{FA} \mathsf{C} \Leftrightarrow \forall_{x\in\Xi_{FA}}.Tr^c([\mathsf{A}]_x) \supseteq Tr^c([\mathsf{C}]_x)$ and as $Det_{FA} \subseteq \Xi_{FA}$ we also have

$\mathsf{A} \sqsubseteq_{FA} \mathsf{C} \Leftrightarrow \forall_{x\in Det_{FA}}.Tr^c([\mathsf{A}]_x) \supseteq Tr^c([\mathsf{C}]_x)$ (1)

From Lemma 1 we then have

$\mathsf{A} \sqsubseteq_{FA} \mathsf{C} \Rightarrow \forall_{x\in Det_{FA}}.D^c([\![[\mathsf{A}]_x]\!]^{FA}_{PPFA}) \supseteq D^c([\![[\mathsf{C}]_x]\!]^{FA}_{PPFA})$.

Then,

$\forall_{x\in Det_{FA}}.D^c([\![[\mathsf{A}]_x]\!]^{FA}_{PPFA}) \supseteq D^c([\![[\mathsf{C}]_x]\!]^{FA}_{PPFA})$

$\forall_{x\in Det_{FA}}.D^c([[\![\mathsf{A}]\!]^{FA}_{PPFA}]_{[\![x]\!]^{FA}_{PPFA}}) \supseteq D^c([[\![\mathsf{C}]\!]^{FA}_{PPFA}]_{[\![x]\!]^{FA}_{PPFA}})$ from Theorem 5

$\forall_{x\in Det_{PPFA}}.D^c([[\![\mathsf{A}]\!]^{FA}_{PPFA}]_x) \supseteq D^c([[\![\mathsf{C}]\!]^{FA}_{PPFA}]_x)$ Lemma 2 part 1(a)

$[\![\mathsf{A}]\!]^{FA}_{PPFA} \sqsubseteq_{PPFA} [\![\mathsf{C}]\!]^{FA}_{PPFA}$ from Definition 9

4. Re: show vA^{FA}_{PPFA} is monotone: $\mathsf{A} \sqsubseteq_{PPFA} \mathsf{C} \Rightarrow vA^{FA}_{PPFA}\mathsf{A} \sqsubseteq_{FA} vA^{FA}_{PPFA}\mathsf{C}$

From $\mathsf{A} \sqsubseteq_{PPFA} \mathsf{C}$ and definitions we have: $\forall_{x\in\Xi_{PPFA}}.D^c([\mathsf{A}]_x) \supseteq D^c([\mathsf{C}]_x)$

as $Det_{PPFA} \subset \Xi_{PPFA}$ we have

$\forall_{x\in Det_{PPFA}}.D^c([\mathsf{A}]_x) \supseteq D^c([\mathsf{C}]_x)$ (2)

For all o in $Tr^c(vA^{FA}_{PPFA}([\mathsf{C}]_x))$ there must exist a d in $D^c([\mathsf{C}]_x)$ such that $o \in dom(d)$ and from (2) we know that d is in $D^c([\mathsf{A}]_x)$ and with $o \in dom(d)$ we can conclude that o in $Tr^c(vA^{FA}_{PPFA}([\mathsf{A}]_x))$ so:

$\forall_{x\in Det_{PPFA}}.Tr^c(vA^{FA}_{PPFA}([\mathsf{A}]_x)) \supseteq Tr^c(vA^{FA}_{PPFA}([\mathsf{C}]_x))$

$\forall_{x\in Det_{PPFA}}.Tr^c([vA^{FA}_{PPFA}\mathsf{A}]_{vA^{FA}_{PPFA}x}) \supseteq Tr^c([vA^{FA}_{PPFA}\mathsf{C}]_{vA^{FA}_{PPFA}x})$ Theorem 5

$\forall_{x\in Det_{FA}}.Tr^c([vA^{FA}_{PPFA}\mathsf{A}]_x) \supseteq Tr^c([vA^{FA}_{PPFA}\mathsf{C}]_x)$ from Lemma 2 part 1(b)

$\forall_{x\in\Xi_{FA}}.Tr^c([vA^{FA}_{PPFA}\mathsf{A}]_x) \supseteq Tr^c([vA^{FA}_{PPFA}\mathsf{C}]_x)$ from Lemma 2 part 3

$vA^{FA}_{PPFA}\mathsf{A} \sqsubseteq_{FA} vA^{FA}_{PPFA}\mathsf{C}$ Definition 3

•

The embedding $[\![\_]\!]^{FA}_{PPFA}$ can be used to add probability to a non-probabilistic finite automata during the stepwise development, i.e. refinement, of a model or specification. This use of Galois connections is nothing new but to the best of our knowledge it is the first time it has been used to allow the introduction of probability part of the way through the development of a process.

8 Conclusions

Others have used the same testing framework to treat probabilistic processes, but in one notable case [9] it was found that many of the expected algebraic results were false according to the testing used. This meant the abandonment of testing as a basis for refinement and a notion of simulation was introduced. We believe that the reason that many of the "sanity checks" turned out to be false for the testing-based refinement in that paper was that the original formalisation of nondeterminism found in non-probabilistic

systems was kept and that this led to problems when probabilistic tests on nondeterministic probabilistic systems were considered.

Instead of abandoning refinement based on testing, we handle nondeterminism in a way that is compatible with probability, rather than using the original formalisations of testing nondeterminism found in non-probabilistic systems.

We also note that, having shown we can (as a (vertical) refinement) move from non-probabilistic models to probabilistic ones (and back again, if we wish), the introduction of probabilities can happen as a design step *during* development of a system via refinement steps. So, we are free to take a very general non-probabilistic specification and, if it turns out to be necessary to do so to deal with some aspects of the specification, introduce probabilities as we make progress towards a more concrete form of the system. We have not yet explored this possibility, but it does introduce another freedom to the developer which might turn out to be useful.

The framework we have introduced in this paper is really only a first step towards a sensible language for specifying systems containing probability. What still needs to be done is to recognise that some sorts of probabilistic choice do not "make sense", i.e. that there are right and wrong places to use such choice. For example, if we have a vending machine with two buttons on, one for tea and one for coffee, it clearly does not make sense to specify the choice here as a probabilistic one—the vending machine would be a very odd one if it allowed me to choose tea only 75% of the time!

On the other hand, it does make sense (though perhaps inventing plausible uses for such a thing might be hard!) to specify a robot which can make choices from a vending machine that offers tea or coffee, where the robot prefers tea over coffee, so it chooses tea 75% of the time.

The difference between these two cases is one of *causality*. The robot's actions cause the vending machine's, and not *vice versa*. So, our specification language would need to allow us to make this distinction and, most helpfully, only allow probabilistic choice to be specified in situations where it makes sense, as in the case of specifying the robot. We have done previous work on adding causality (back) into process algebras, and the work presented here forms the basis for a probabilistic causal process algebra (CPA) [12], or for a probabilistic language for interactive branching processes (IBPs) [10] which we have also talked about before, which forms the subject of another paper yet to be published.

A final interesting point to note is that, because we can always migrate probabilities on actions right up the probabilities on start states, we have a normal form for our automata. In this form, the only place that probabilities appear is on the start states (so the only non-trivial probabilistic distribution over states is the start-state distribution). This makes it very clear that one needs only one roll (of dice with enough faces) in order to conduct a probabilistic computation.

References

[1] Hennessy, M.: Algebraic Theory of Processes. The MIT Press (1988)

[2] López, N., Núñez, M.: An overview of probabilistic process algebras and their equivalences. In Baier, C., Haverkort, B.R., Hermanns, H., Katoen, J.P., Siegle, M., eds.: Validation of Stochastic Systems. Volume 2925 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2004) 283–298 doi:10.1007/978-3-540-24611-4_3

[3] Reeves, S., Streader, D.: Unifying state and process determinism. Technical report, University of Waikato, http://hdl.handle.net/10289/1001 (2004)

[4] Baeten, J.C.M., Weijland, W.P.: Process Algebra. Cambridge Tracts in Theoretical Computer Science 18 (1990)

[5] Winskel, G., Nielsen, M.: Models for concurrency. Technical Report DAIMI PB 429, Computer Science Dept. Aarhus Universty (1992)

[6] Reeves, S., Streader, D.: Guarded operations, refinement and simulation. In: Proc Fourteenth BAC-FACS Refinement Workshop (REFINE 2009). Volume 259 of Electronic Notes in Theoretical Computer Science., Eindhoven, The Netherlands, Elsevier (2009) 177–191 doi:10.1016/j.entcs.2009.12.024

[7] Hoare, C.: Communicating Sequential Processes. Prentice Hall International Series in Computer Science (1985)

[8] Reeves, S., Streader, D.: Atomic Components. In Liu, Z., Araki, K., eds.: Theoretical Aspects of Computing - ICTAC 2004: First International Colloquium. Volume 3407 of Lecture Notes in Computer Science., Springer-Verlag (2004) 128–139

[9] Deng, Y., van Glabbeek, R., Hennessy, M., Morgan, C., Zhang, C.: Remarks on testing probabilistic processes. Electron. Notes Theor. Comput. Sci. **172** (2007) 359–397 doi:10.1016/j.entcs.2007.02.013

[10] Reeves, S., Streader, D.: Contexts, refinement and determinism. Science of Computer Programming, DOI: 10.1016/j.scico.2010.11.011 (2010) doi:10.1016/j.scico.2010.11.011

[11] Taylor, P.: Practical Foundations of Mathematics. Cambridge University Press (1999) Cambridge studies in advanced mathematics 59.

[12] Reeves, S., Streader, D.: Causal process algebra: always getting the right drink from a coffee machine. In: Proc EPSRC RefineNet Refinement Workshop, Seventh International Conference on Formal Engineering Methods (ICFEM 2005). (2005) 1–14