Programming Research Group

PROBABILISTIC PREDICATE TRANSFORMERS

Carroll Morgan, Annabelle McIver, Karen Seidel and JW Sanders

PRG-TR-4-95

Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD

Probabilistic predicate transformers

Carroll Morgan*, Annabelle McIver,
Karen Seidel and JW Sanders†

Draft of 9 February 1995

Abstract

Predicate transformers facilitate reasoning about imperative programs, including those exhibiting demonic non-deterministic choice. Probabilistic predicate transformers extend that facility to programs containing probabilistic choice, so that one can in principle determine not only whether a program is guaranteed to establish a certain result, but also its probability of doing so.

We bring together independent work of Claire Jones and Jifeng He, showing how their constructions can be made to correspond; from that link between a predicate-based and a relation-based view of probabilistic execution we are able to propose 'probabilistic healthiness conditions', generalising those of Dijkstra for ordinary predicate transformers.

The associated calculus seems suitable for exploring further the rigorous derivation of imperative probabilistic programs.

Keywords: Probability, predicate transformers, non-determinism, verification, refinement, imperative programming, program derivation, Galois connection.

*Morgan was partially supported during this work by the Dutch *Specification and Transformation of Programs* (STOP) project.

†All authors are members of the Programming Research Group at the University of Oxford: {*carroll,anabel,karen,jeff*}*@comlab.ox.ac.uk*. McIver and Seidel are supported by the SERC.

1 Introduction

We begin by considering deterministic, non-deterministic and probabilistic programs in turn.

To express that the deterministic program $n := 1$ is guaranteed to establish $n = 1$, using ordinary predicate transformers [4], we write

$$wp.(n := 1).(n = 1) \quad = \quad true\ .$$

The non-deterministic program $n := 1 \sqcap n := 2$ chooses demonically between the two alternatives $n := 1$ and $n := 2$, and is guaranteed to establish $n = 1 \vee n = 2$; but we can say nothing about whether it will establish $n = 1$ or $n = 2$ separately. The weakest precondition formulation

$$wp.(n := 1 \sqcap n := 2).(n = 1) \ = \ false \ = \ wp.(n := 1 \sqcap n := 2).(n = 2)$$

expresses that pessimistic view (the usual): there are no conditions under which termination establishing $n = 1$ (or $n = 2$) is guaranteed.

The probabilistic program $n := 1 \ {}_{\frac{1}{2}}\oplus\ n := 2$ also chooses between two alternatives, and it too is guaranteed to establish $n = 1 \vee n = 2$. Here however we can say something about whether it will establish $n = 1$ or $n = 2$ separately: with a suitable generalisation of *wp* we will find that

$$wp.(n := 1 \ {}_{\frac{1}{2}}\oplus n := 2).(n = 1) \ = \ \frac{1}{2} \ = \ wp.(n := 1 \ {}_{\frac{1}{2}}\oplus n := 2).(n = 2)\ ,$$

expressing that termination establishing $n = 1$ (or $n = 2$) is guaranteed with probability at least $1/2$.

The origin of the probabilistic predicate transformers we present here is Claire Jones' thesis [11], where they are given as an example of Jones and Plotkin's general technique for introducing probability into a semantic domain [12]. In [11], however, only deterministic programs are considered; and the underlying operational model is therefore one of functions, from initial to final state.

Later, and independently of Jones, Jifeng He proposed a relational (rather than functional) model of probabilistic execution [9] partly in response to some surprising characteristics of non-determinism in [19]: both propose a 'convex and up-closed' set of distributions as a model in which non-determinism and probabilistic choice can coexist. (Non-determinism as sets of distributions is suggested in [15] also.)

The contribution of this paper is to show that Jones' and He's models go together — that the relations of He provide a more general operational semantics underlying the predicate transformers of Jones, leading in particular to a simple treatment of non-determinism and probability together. In the standard case such a correspondence is well known, and it allows the determination of 'healthiness' (well-formedness) conditions for predicate transformers that characterise collectively the images of relational programs in the richer predicate-transformer space: strictness, monotonicity, conjunctivity and continuity.

In the probabilistic case, a similar correspondence allows us to discover probabilistic healthiness conditions.

Our results are presented as follows: in Sec. 6 we introduce the correspondence between He's and Jones' models; in Sec. 7 we propose probabilistic healthiness conditions; in Sec. 8 we show that those conditions characterise 'ordinary programs' exactly; and in Sec. 9 we extend Jones' work to include non-determinism. Sec. 10 discusses probabilistic termination, and recursion.

Earlier sections summarise He's and Jones' approaches, on which our work is based; and example programs are given in Sections 2, 4, 9, 10 and 11.

We use infix dot for function application (thus $f.x$, not $f(x)$), and the syntax

$$(\mathcal{Q}s{:}\, S \mid range \cdot expr)$$

for $\mathcal{Q}$-quantification (for example $\mathcal{Q}$ as $\forall$, $\exists$, λ or set comprehension) of the values *expr* formed when s ranges over those elements of S satisfying *range*. A glossary is given in App. B.

2 Deterministic probability

We now consider a simple operational model for deterministic probabilistic behaviour.

A deterministic imperative program begins execution in an *initial* state and terminates (if it does) in a *final* state. We take *deterministic* to mean that for every initial state from which termination is guaranteed there is exactly one corresponding final state; from an initial state that can lead to non-termination, however, any final state is possible as well.[1]

That operational view is expressed as follows: given a set S of states, the *state space*, a (non-probabilistic) deterministic program denotes a total function of type $S \rightarrow S_{\perp}$, where we write $S_{\perp}$ for the set $S \cup \{\perp\}$, so that non-termination can be indicated when appropriate by an (improper) final state $\perp$. We call non-probabilistic programs *standard*.

A probabilistic program, in constrast, takes an (initial) state to a (final) probabilistic distribution over S rather than to some element of $S_{\perp}$. We use the 'evaluations' of [11, Sec. 3.9] to describe those distributions (though we continue to call them 'distributions'), specialising Jones' definition to S regarded as a flat domain:

Definition 2.1 For state space S, the set of *distributions* over S is

$$\overline{S} \quad := \quad \{D\colon S \rightarrow [0,1] \mid \sum D \leq 1\} \ ,$$

the set of functions from S into the closed interval of reals $[0,1]$ that sum to no more than 1. (We write $\sum D$ to abbreviate $\sum_{s:S} D.s$.) $\Box$

Thus a distribution over S assigns a probability to every element individually of S, and those probabilities must sum to no more than 1.[2]

Def. 2.1 differs slightly from the usual 'probability function over an event space' of elementary probability (*eg.* [7]). It is more specific in that by assigning probabilities to individual elements of S we are taking the (discrete) event space comprising all subsets of S, which means for example that we cannot describe a 'uniform' distribution if S is infinite: we are therefore unable to treat programs like

$$x\!:= \text{`some value chosen uniformly from } [0,1]\text{'} \ .$$

It is more general in that the restriction $\sum D \leq 1$ is not an equality, so that for D in $\overline{S}$ the difference

$$1 - \sum D$$

may be regarded as the probability of 'no state at all' — a convenient treatment of non-termination that allows us to forego $\perp$.

There is a partial order over $\overline{S}$, inherited pointwise from $[0,1]$:

[1] Some writers call that *pre-determinism*: 'deterministic if terminating'.

[2] In the terminology of [13, p.331] such distributions would be called *discrete sub-probability measures*.

Definition 2.2 For $D, D' \colon \overline{S}$,

$$D \sqsubseteq D' \quad := \quad (\forall s \colon S \cdot D.s \leq D'.s) \; .$$

$\square$

The least element of $\overline{S}$ is $(\lambda s \colon S \cdot 0)$, which we write $\mathbf{o}$; and D in $\overline{S}$ is maximal when $\sum D = 1$, in which case D indicates certain termination.

Thus we define a space of probabilistic deterministic programs together with a refinement order:

Definition 2.3 For state space S the space of deterministic probabilistic programs over S is defined

$$\mathcal{D}S \quad := \quad (S \to \overline{S}, \sqsubseteq) \; ,$$

where for programs d, d' in $S \to \overline{S}$ we define $\sqsubseteq$ pointwise (again):

$$d \sqsubseteq d' \quad := \quad (\forall s \colon S \cdot d.s \sqsubseteq d'.s) \; .$$

The order $\sqsubseteq$ of $\mathcal{D}S$ is called the *refinement* order. $\square$

It is easy to see that $\mathcal{D}S$ is a complete partial order; and its least element is the nowhere-terminating program

$$\mathbf{abort}.s := \mathbf{o} \quad \text{or equivalently} \quad \mathbf{abort}.s.s' := 0 \; , \tag{1}$$

for initial state s and final state s'. (Recall that **abort** is deterministic in our sense, because it is nowhere-terminating.) We see thus that for every program d in $\mathcal{D}S$ we have $\mathbf{abort} \sqsubseteq d$.

In general, if $\sum d.s = 1$ then $d.s.s'$ is the probability that d takes s to s'; but if $\sum d.s < 1$ then $d.s.s'$ is only a lower bound for that probability, since the aborting component of weight $1 - \sum d.s$ is unpredictable. Thus for **abort** itself, where that lower bound is 0, we are told nothing about the probability of reaching any particular final s'.

For state s in S we write $\overline{s}$ for the distribution 'certainly s':

Definition 2.4 For state s the *point distribution* at s is defined

$$\overline{s}.s' \quad := \quad \begin{array}{ll} 1 & \text{if } s = s' \\ 0 & \text{otherwise.} \end{array}$$

$\square$

Thus a standard deterministic program can be lifted into the probabilistic space as follows:

Definition 2.5 For every standard deterministic program t in $S \to S_\perp$ there is a corresponding probabilistic program in $\mathcal{D}S$, defined

$$\begin{array}{lll} \overline{t}.s & := & \overline{t.s} \quad \text{if } t.s \neq \perp \\ & & \mathbf{o} \quad\ \ \text{otherwise,} \end{array}$$

where s is an arbitrary (initial) state. Equivalently we could define (for arbitrary final state s')

$$\begin{array}{lll} \overline{t}.s.s' & := & 1 \quad \text{if } t.s = s' \\ & & 0 \quad \text{otherwise,} \end{array}$$

noting that s, s' are restricted to proper values (not $\perp$). □

A properly probabilistic program over state space $\mathbb{N}$ is for example

$$n:=\ 1\ {}_{\frac{1}{3}}\!\oplus n:=\ 2\ , \tag{2}$$

where in general ${}_p\oplus$ chooses its left,right operand with probability $p, 1-p$ respectively. The meaning of (2) is $d\colon \mathcal{D}\mathbb{N}$, where

$$\begin{array}{lll} d.n.n' & := & 1/3 \quad \text{if } n' = 1 \\ & & 2/3 \quad \text{if } n' = 2 \\ & & 0 \qquad \text{otherwise.} \end{array}$$

(The absence of n on the right reflects only that (2) is insensitive to its initial state.)

We do not give here the definitions of program operators for $\mathcal{D}S$, since we treat them later in a more general setting (Defs. 5.5–5.7). But note for example that if we define

$$n:=\ 1\ {}_{\frac{1}{3}}\!\oplus \mathbf{abort} \tag{3}$$

by d' where

$$\begin{array}{lll} d'.n.n' & := & 1/3, \quad \text{if } n' = 1 \\ & & 0, \qquad \text{otherwise,} \end{array}$$

then we have $\mathbf{abort} \sqsubset d' \sqsubset d$, thus illustrating both non-strictness of ${}_{\frac{1}{3}}\!\oplus$ and proper (non-trivial) refinement.

3 Expected values

Expected values of random variables are the basis for probabilistic predicate transformers.

A *random variable* α is a real-valued function over the sample space[3] (for us, S) and, given a probability distribution D (from $\overline{S}$) we can define the expected value of α as follows [7]:

Definition 3.1 For random variable $\alpha: S \rightarrow \mathbb{R}$ and distribution $D: \overline{S}$ the *expected value of α over D* is

$$\int_D \alpha \quad := \quad \sum_{s:S} (\alpha.s \times D.s) \; .$$

□

(We use the integral notation for compatibility with expectations in general.[4])

With Def. 3.1 we can write the expectation of the square of the final value delivered by a program $d: \mathcal{D}\mathbb{N}$, if executed in initial state n, as

$$\int_{d.n} n^2 \,\mathrm{d}n \;, \tag{4}$$

where we read $(n^2 \,\mathrm{d}n)$ as the function $(\lambda n \cdot n^2)$ over the natural numbers. (Note that n in $d.n$ is free (the initial state), but n in $(n^2 \,\mathrm{d}n)$ is bound.) Applying (4) to Program (2) gives us $(1/3)1^2 + (2/3)2^2$, or expectation 3 for the square of the final values delivered.[5]

---

[3] That simple definition is possible because of the discrete nature of our distributions $\mathcal{D}S$; the functions are 'measurable', satisfying for example [7, p.22 Condition 2] trivially.

[4] For example, the expectation (or average) of the function $f: \mathbb{R} \rightarrow \mathbb{R}$ over the interval $[a, b]$ is given in analysis by

$$\frac{1}{b-a} \times \int_a^b f(x)\,\mathrm{d}x \;,$$

which in the above style one could write $\int_{\langle a,b \rangle} f$, inventing the notation $\langle a, b\rangle$ to denote a uniform distribution on $[a, b]$. We are in fact dealing with integration over measures; but for ease of calculation we write $\int_\mu f$ rather than the $\int f \mathrm{d}\mu$ of [11].

[5] In practice we consider only non-negative random variables, so that even in the presence of non-termination we get a 'minimum' expectation.

More significantly, Def. 3.1 can be used to apply a probabilistic program to a distribution of initial states (rather than only to a 'definite' single state), for example to the (intermediate) distribution yielded by the program immediately before it in a sequential composition. Following [11, Sec. 4.2] we have

Definition 3.2 For $d{:}\,\mathcal{D}S$, (initial) distribution $I{:}\,\overline{S}$ and (final) state $s'{:}\,S$ we define $d^\dagger$, an element of $\overline{S} \rightarrow \overline{S}$, as follows:

$$d^\dagger.I.s' \quad := \quad \int_I d.s.s' \,\mathrm{d}s \;.$$

□

Thus we may if we wish regard probabilistic deterministic programs as homogeneous functions (from $\overline{S}$ to itself), taking initial to final distributions — the distributions then represent 'probabilistic states', as for example in [13], and in execution the program moves from one probabilistic state to the next.[6]

The specialisation to our earlier view ($S \rightarrow \overline{S}$) is obtained for initial state s by taking the point distribution $\overline{s}$ concentrated there; we then observe that from Def. 3.2 and Def. 3.1 we have

$$d^\dagger.\overline{s}.s' \quad = \quad \int_{\overline{s}} d.s.s' \,\mathrm{d}s \quad = \quad d.s.s' \;,$$

so that $d.s$ is recovered as $d^\dagger.\overline{s}$.

The homogeneous view allows for example a simple definition for sequential composition of the deterministic programs d, d' — it is the Kleisli-like $d{\circ}d'^\dagger$ — but we postpone consideration of program operators until Sec. 5, where we treat nondeterminism as well.

4 Probabilistic program logic

The logic of [11, Chap. 7] treats deterministic probabilistic programs at the level of Hoare triples [10] and weakest preconditions [4]. We summarise it in this section without relying on determinism in the presentation.

[6] But our programs $d^\dagger$ satisfy [13, Thm. 6.1] by construction: they are completely determined by their behaviour on the point distributions $\overline{s}$ of $\overline{S}$.

A standard program t (deterministic or not) is said to satisfy

$$\begin{array}{rll} & \{pre\}\ t\ \{post\} & [10] \\ \text{or equivalently} & pre \Rightarrow wp.t.post & [4], \end{array}$$

for predicates $pre, post$ over the state space S, if whenever execution of t begins in a state satisfying pre it will terminate in a state satisfying $post$.

Following [11, Chap. 7], for the probabilistic case we use random variables α, β rather than predicates $pre, post$, and expected values rather than membership. Viewing a probabilistic program h as moving from an initial to some final distribution, thus taking the homogeneous view,

$$\begin{array}{rl} & \{\alpha\}\ h\ \{\beta\} \\ \text{or equivalently} & \alpha \leq wp.h.\beta \end{array}$$

means

> for every pair of distributions $I, F{:}\,\overline{S}$ such that h can take (initial) I to (final) F,
>
> $$\int_I \alpha \quad \leq \quad \int_F \beta\ . \tag{5}$$

Thus whereas in the standard case 'the truth value of the assertions must not decrease as execution proceeds', in the probabilistic case it is the expectations that must not decrease.

To lift the standard logic into the probabilistic we consider pre and $post$ as subsets of S, and form random variables

$$\begin{array}{rcl} \alpha & := & \chi_{pre} \\ \beta & := & \chi_{post} \end{array}$$

where for general subset T of S we write χ_T for the *characteristic function* of T. The standard $\{pre\}\ t\ \{post\}$ then says about t just what the probabilistic

$$\{\chi_{pre}\}\ h\ \{\chi_{post}\}$$

says about h; for if we take the point distributions $\overline{s}, \overline{s'}$ corresponding to standard initial and final states s, s' then (5) becomes

> for every pair of states $s, s'{:}\,S$, such that h can take s to s',
>
> $$\int_{\overline{s}} \chi_{pre} \quad \leq \quad \int_{\overline{s'}} \chi_{post}\ ,$$

standard	expression	probabilistic
true iff x and y are equal	$x = y$	1 if x equals y, 0 if they differ
true iff α is *false*	$\neg\alpha$	$1 - \alpha$
true iff α or β is *true*	$\alpha \vee \beta$	$\alpha \sqcup \beta$, the maximum of α and β .

Figure 1: Idioms for probabilistic predicates

$$\begin{array}{rcl}
wp.\mathbf{skip}.\beta & := & \beta \\
wp.(x{:=}\ E).\beta & := & \beta[x{:=}\ E] \\
wp.(h;h').\beta & := & wp.h.(wp.h'.\beta) \\
wp.(\mathbf{if}\ B\ \mathbf{then}\ h\ \mathbf{else}\ h'\ \mathbf{fi}).\beta & := & B \times wp.h.\beta + (\neg B) \times wp.h'.\beta \\
wp.(h\ {}_p\!\oplus\ h').\beta & := & p \times wp.h.\beta + (1-p) \times wp.h'.\beta
\end{array}$$

The notation $\beta[x{:=}\ E]$ denotes syntactic replacement of all free occurrences of variable x in predicate β by expression E.

Figure 2: Probabilistic predicate semantics

which simplifies (Def. 3.1) first to $\chi_{pre}.s \leq \chi_{post}.s'$, and finally to

$$s \in pre \quad \Rightarrow \quad s' \in post \ .$$

We call random variables over the state (probabilistic) predicates, when used like α, β above for reasoning about programs: they are our preconditions and postconditions. And we call such a predicate *standard* if it takes values in $\{0,1\}$ only (equivalently, if it is χ_T for some T a subset of S.)

In practice, probabilistic predicates are written as real-valued expressions over the program variables, punning as indicated in Fig. 1 between Boolean values and $\{0,1\}$ where convenient. With those conventions we give in Fig. 2 the weakest precondition semantics of a simple (deterministic) probabilistic language [11, p. 146 simplified].

Returning to Program (2), for example, we have from Fig. 2

$$\begin{array}{ll}
 & wp.(n{:=}\ 1\ {}_{\frac{1}{3}}\!\oplus\ n{:=}\ 2).(n=1) \\
= & 1/3 \times wp.(n{:=}\ 1).(n=1) + 2/3 \times wp.(n{:=}\ 2).(n=1) \\
= & 1/3 \times (1=1) + 2/3 \times (2=1) \\
= & 1/3 \times 1 + 2/3 \times 0 \qquad \text{punning between } \textit{true} \text{ and } 1, \textit{false} \text{ and } 0 \\
= & 1/3\ ,
\end{array} \tag{6}$$

and similarly $wp.(n{:=}\ 1\ {}_{\frac{1}{3}}\!\oplus\ n{:=}\ 2).(n=2)$ is $2/3$. Finally,

$$
\begin{array}{ll}
 & wp.(n{:=}\ 1\ {}_{\frac{1}{3}}\!\oplus\ n{:=}\ 2).(n=1 \vee n=2) \\
= & 1/3 \times (1 \sqcup 0) + 2/3 \times (0 \sqcup 1) \\
= & 1\ .
\end{array}
\tag{7}
$$

Note that the final values $1/3$, $2/3$ and 1 are (constant) expressions over the state variables, and should be regarded (when necessary) as constant functions $(\lambda n \cdot 1/3)$ *etc.*

An interpretation of (6) and (7) is that Program (2) has (at least) probability $1/3$ of establishing $n = 1$, and probability 1 (thus certainty) of establishing $n = 1 \vee n = 2$. The general reasoning leading to those conclusions is as follows.

Let h be a probabilistic program, *post* a set of (desired final) states, α a probabilistic precondition and s an (initial) state. Then

$$\{\alpha\}\ h\ \{\chi_{post}\} \tag{8}$$

means that for any initial distribution I, and final distribution F produced by h from it,

$$\int_I \alpha \quad \leq \quad \int_F \chi_{post}\ .$$

Taking I to be $\overline{s}$ in particular, and referring to Def. 3.1, we have then

$$\alpha.s \quad \leq \quad \sum_{s':post} F.s'\ , \tag{9}$$

and thus that the value of α at the initial state s bounds below the probability that the final state s' lies in *post*, no matter what final F may be chosen by h executed from s. Since $wp.h.\beta$ is in general the weakest (largest) α such that $\{\alpha\}\ h\ \{\beta\}$, we conclude that

$$wp.h.\chi_{post}.s \tag{10}$$

is the largest $\alpha.s$ satisfying (9) for all such F, and so is the minimum probability, over all possible final distributions, that h takes initial s into final *post*. (That 'minimum' view will prove to be convenient later, when we consider non-determinism explicitly: demonic choice strives to make the probability of reaching *post* as low as possible.)

More generally than (8), however, the postconditions will not be standard, and then the (properly) probabilistic predicates will work backwards through sequential composition as in this example:

$$
\begin{array}{ll}
 & wp.(x{:=}\ 1\ {}_{\frac{1}{2}}\!\oplus\ x{:=}\ 2;\ \ y{:=}\ 1\ {}_{\frac{1}{2}}\!\oplus\ y{:=}\ 2).(x \leq y) \\
= & wp.(x{:=}\ 1\ {}_{\frac{1}{2}}\!\oplus\ x{:=}\ 2).(wp.(y{:=}\ 1\ {}_{\frac{1}{2}}\!\oplus\ y{:=}\ 2).(x \leq y)) \\
= & wp.(x{:=}\ 1\ {}_{\frac{1}{2}}\!\oplus\ x{:=}\ 2).((x \leq 1)/2 + (x \leq 2)/2) \qquad (11) \\
= & (1 \leq 1)/4 + (1 \leq 2)/4 + (2 \leq 1)/4 + (2 \leq 2)/4 \\
= & 1/4 + 1/4 + 0/4 + 1/4 \\
= & 3/4\ .
\end{array}
$$

Note in particular that at line (11) the 'intermediate' postcondition

$$(x \leq 1)/2 + (x \leq 2)/2$$

is not standard, yet the overall result may still be interpreted as at (8) and (10) above: the program establishes $x \leq y$ with probablity at least 3/4. (In fact it is exactly 3/4, because the program is deterministic and terminating.) In [11, Sec. 7.8], Claire Jones relates the operational view of Sec. 2 to the axiomatic view (this section), showing soundness and completeness. For deterministic probabilistic program h and initial state s, the connection she gives is

$$wp.h.\beta.s = \int_{h.s} \beta \quad \text{and} \quad h.s.s' = wp.h.\mathcal{X}_{\{s'\}}.s\ .$$

Our function wp of Sec. 6 generalises that.

5 Non-determinism and relations

In [9] Jifeng He proposes a relational model for imperative probability more general than in Sec. 2 (from [11, Ch. 7]): a program executed in an initial state produces not a (single) final distribution but rather a set of them, and the plurality of the set represents non-determinism. We show that He's construction, although devised independently of Jones' work, with slight modification provides a model for her program logic and indeed motivates its extension to include non-determinism. In this section we explain He's model, and in the following we give the connection between it and Jones' probabilistic program logic.

For ease of presentation we assume from this point on that the state space S is of finite cardinality.[7]

[7] In [20] it is shown that our results continue to hold in the infinite case, slightly adjusted.

Not every set of distributions is appropriate as the result of a probabilistic non-deterministic program; we consider only non-empty, up-closed, convex and Cauchy-closed sets, constraints which we now discuss.

Non-emptiness is imposed to avoid 'miracles', or 'infeasible programs' [23, 17, 22] which, although useful, would complicate our presentation here.

For up-closure, we recall from Def. 2.2 that $\overline{S}$ is a *cpo*; thus

Definition 5.1 A subset $\mathcal{F}$ of $\overline{S}$, a set of distributions, is *up-closed* if it is closed under refinement of its elements — if for all $F, F' \colon \overline{S}$,

$$F \in \mathcal{F} \wedge F \sqsubseteq F' \quad \Rightarrow \quad F' \in \mathcal{F} \ .$$

$\square$

We insist on up-closure so that refinement (of nondeterministic programs) is expressed by reverse subset inclusion of their result sets: program h is refined by h' if for every initial state s we have $h'.s \subseteq h.s$.

Recall for example the program **abort**. The standard **abort** takes every s to $\perp$; in the deterministic probabilistic model of Sec. 2 it takes every s to the constant distribution $\mathbf{o}$; and so in our current model it takes every s to the up-closure of $\{\mathbf{o}\}$. But $\mathbf{o}$ is the least element of $\overline{S}$ — and so its up-closure is $\overline{S}$ itself, allowing every possible behaviour just as one expects from **abort**.

Convexity of a set of distributions is defined as follows:

Definition 5.2 A set $\mathcal{F}$ of distributions is *convex* if for every $F, F' \colon \mathcal{F}$ and $p \colon [0,1]$ we have $F \;{}_p{\oplus}\; F' \in \mathcal{F}$ also, where

$$(F \;{}_p{\oplus}\; F').s \quad := \quad p \times F.s + (1-p) \times F'.s \ .$$

$\square$

We insist on convexity so that non-determinism is refined by any probabilistic choice. Consider for example the program

$$n := 1 \sqcap n := 2 \ , \tag{12}$$

where we use $\sqcap$ to indicate nondeterministic (demonic) choice.[8] The result set of distributions will contain the two point-distributions $\overline{1}, \overline{2}$, and

[8]The standard definition is $wp.(t \sqcap t').post := wp.t.post \wedge wp.t'.post$.

by convexity will contain also the distributions $\overline{1}\ {}_p\!\oplus \overline{2}$ for any p: $[0,1]$. Thus our notion of refinement allows Program (12) to be refined to

$$n{:=}\ 1\ \ {}_p\!\oplus\ \ n{:=}\ 2\ ,$$

for any probability p.

For Cauchy-closure, we note that our distributions F in $\overline{S}$ are points in Euclidean space, where each axis corresponds to an element s of S and the s-coördinate of F is $F.s$. We have then

Definition 5.3 A set of distributions over S is *Cauchy-closed* if as a subset of $\mathbb{R}^N$ it is closed in the usual (Euclidean) sense, where N is the (finite) cardinality of S. □

We now give the relational model for nondeterministic probabilistic programs.

Definition 5.4 Given a (finite) state space S, the set of non-empty, up-closed, convex and Cauchy-closed subsets of $\overline{S}$ is written $\mathcal{C}S$. The *cpo* of nondeterministic probabilistic programs over S is then

$$\mathcal{H}S \quad := \quad (S \to \mathcal{C}S, \sqsubseteq)\ ,$$

where for h, h': $\mathcal{H}S$

$$h \sqsubseteq h' \quad := \quad (\forall s{:}\, S \cdot h.s \supseteq h'.s)\ .$$

□

Note that Cauchy-closure in $\mathcal{C}S$ ensures that $\mathcal{H}S$ is indeed a *cpo*, for the infinite intersection of a $\supseteq$-directed collection of non-empty closed subsets of $[0,1]^N$ cannot be empty. The other conditions on $\mathcal{C}S$ are preserved trivially by arbitrary intersection.

For a full account of the structure of $\mathcal{H}S$ we refer to [9]; here we discuss only probabilistic choice ${}_p\!\oplus$, non-deterministic choice $\sqcap$ and sequential composition. For well-definedness (preservation of the closure conditions) we refer to [9] for the first two; we have added Cauchy-closure ourselves, and it is discussed following the definitions.

Probabilistic choice between programs is formed by taking the appropriate combinations of the elements of the result sets.

Definition 5.5 For two programs $h, h' {:} \mathcal{H}S$, their p-probabilistic combination is defined

$$(h \;{}_p\!\oplus\; h').s \quad := \quad \{F{:}\, h.s; F'{:}\, h'.s \cdot F \;{}_p\!\oplus\; F'\} \; .$$

□

Non-deterministic choice is formed simply by taking all possible probabilistic combinations. Thus

Definition 5.6 For two programs $h, h' {:} \mathcal{H}S$, their non-deterministic combination is defined

$$(h \sqcap h').s \quad := \quad (\cup p{:}\, [0,1] \cdot (h \;{}_p\!\oplus\; h').s) \; .$$

□

Note that the extreme values 1,0 for p ensure that $h \sqcap h' \sqsubseteq h$ and $h \sqcap h' \sqsubseteq h'$.

Finally, for sequential composition we have

Definition 5.7 For two programs $h, h' {:} \mathcal{H}S$, their sequential composition is defined

$$(h; h').s \quad := \quad \{F{:}\, h.s; d{:}\, \mathcal{D}S \mid h' \sqsubseteq d \cdot d^{\dagger}.F\} \; .$$

□

The sequential composition is formed by taking all possible 'intermediate' distributions F that h can deliver from s and for each continuing with all possible deterministic refinements d of h'.

For well-definedness we are obliged to show that for Cauchy-closed arguments each of the above three operators produces a Cauchy-closed result. The basic argument is the same in each case: a continuous function between Euclidean spaces is applied to a closed and bounded (hence compact) set, and so its image is closed. We consider Def. 5.6 as an example:

Lemma 5.8 If subsets $\mathcal{F}, \mathcal{F}'$ of $\overline{S}$ are Cauchy-closed then so is

$$(\cup p{:}\, [0,1] \cdot (h \;{}_p\!\oplus\; h').s) \; .$$

Proof: By Def. 5.5 the set concerned may be written

$$\{F: \mathcal{F}; F': \mathcal{F}'; p: [0,1] \cdot F \;{}_p{\oplus}\; F'\} \ ,$$

which is the image in $\mathbb{R}^N$ through a continuous function of the closed and bounded subset

$$\mathcal{F} \times \mathcal{F}' \times [0,1]$$

of $\mathbb{R}^{2N+1}$, where N is the cardinality of S. □

6 Regular predicate transformers

6.1 The relational and predicate models

A *relational* model for standard non-deterministic programs over S is

$$S \leftrightarrow S_\perp \ ,$$

where a standard program t relates initial s to final s' just when executing t from initial state s can yield final state s'. A convenient constraint on elements t of $S \leftrightarrow S_\perp$ is that whenever s is taken by t to $\perp$ then it is taken to all of $S_\perp$ (thus allowing simple $\supseteq$ to be the refinement order).

If an element of $S \leftrightarrow S_\perp$ relates every initial state to at least one final state, we say that it is *total*; and if it relates every initial state to at most finitely many final states, we say that it is *image finite*.[9] We write $[S \leftrightarrow S_\perp]$ for the total and image-finite elements of $S \leftrightarrow S_\perp$.

A probabilistic analogue for $[S \leftrightarrow S_\perp]$ is $\mathcal{H}S$.

The *predicate-transformer* model for standard non-deterministic programs is

$$\mathbb{P}S \leftarrow \mathbb{P}S \ ,$$

where we write the functional arrow backwards to emphasise that such programs map sets of final states (postconditions) to sets of initial states (preconditions). A probabilistic analogue for $\mathbb{P}S \leftarrow \mathbb{P}S$ is suggested by the program logic of Sec. 4:

[9]That latter happens automatically under our present assumption that S is finite.

	standard	probabilistic
relational	$[S \leftrightarrow S_\perp]$	$\mathcal{H}S$
predicate	$\mathbb{P}S \leftarrow \mathbb{P}S$	$\mathcal{J}S$

Figure 3: Nomenclature for the four models.

Definition 6.1 For state space S, the set of probabilistic predicates over S is defined

$$\mathcal{P}S \quad := \quad (S \to \mathbb{R}_\geq, \leq) \ ,$$

where $\mathbb{R}_\geq$ is the non-negative reals, and $\leq$ is inherited pointwise from $\mathbb{R}_\geq$. The probabilistic predicate transformer model for programs is then

$$\mathcal{J}S \quad := \quad (\mathcal{P}S \leftarrow \mathcal{P}S, \sqsubseteq) \ ,$$

where the *refinement* order $\sqsubseteq$ over programs is derived pointwise from the order on $\mathcal{P}S$. $\square$

We refer to the four models as shown in Fig. 3; thus $\mathcal{J}S$ for example is the probabilistic predicate model.

The standard relations can be embedded in the standard predicate transformers $\mathbb{P}S \leftarrow \mathbb{P}S$ such that the image of $S \leftrightarrow S_\perp$ is characterised by *conjunctivity*:

> a predicate transformer $p\colon \mathbb{P}S \leftarrow \mathbb{P}S$ is the embedding of some relation in $S \leftrightarrow S_\perp$ iff for all postconditions $post, post'$ in $\mathbb{P}S$ we have
>
> $$p.(post \cap post') \quad = \quad p.post \cap p.post' \ .$$

Thus the standard predicate model is richer than the relational precisely because not all of its programs are conjunctive.[10]

In the remainder of this section we construct an embedding for the probabilitic models.

[10]The extra constraints on $[S \leftrightarrow S_\perp]$ induce strictness and continuity on the corresponding predicate transformers.

6.2 Embedding the relational model in the predicate model

Recall from Sec. 4 that for program $h\colon \mathcal{H}S$, initial s and every possible F that h may produce from s, we must have

$$wp.h.\beta.s \quad \leq \quad \int_F \beta \; . \tag{13}$$

(Just specialise (5) to $I := \overline{s}$ and $\alpha := wp.h.\beta$.) Noting that $wp.h.\beta$ should be as large as possible while satisfying (13) for all F in $h.s$, we propose

Definition 6.2 The injection $wp\colon \mathcal{H}S \to \mathcal{J}S$ is defined

$$wp.h.\beta.s \quad := \quad (\sqcap F\colon h.s \cdot \int_F \beta) \; ,$$

for program $h\colon \mathcal{H}S$, predicate $\beta\colon \mathcal{P}S$ and state $s\colon S$. □

That (probabilistic) wp is indeed an injection is a consequence of Lem. 8.2 below.

That we take the minimum $\sqcap$ over result distributions F is related to our demonic view of non-determinism: the demon resolves non-determinism in a way that makes the weakest precondition as small (as strong) as possible. Note that well-definedness of the minimum is guaranteed by non-emptiness of $h.s$, which is why we impose it; a more general treatment is possible however if ∞ is allowed as a possible value for $\beta.s$.

It is instructive to specialise Def. 6.2 to the standard case. Let β there be χ_{post} for arbitrary standard postcondition *post* and suppose h is standard, so that $h.s$ is the closure of some set of point distributions; then

$$
\begin{array}{ll}
 & wp.h.\chi_{post}.s \\
= & (\sqcap F\colon h.s \cdot \int_F \chi_{post}) \\
= & \hfill h \text{ is a standard program: see below} \\
 & (\sqcap s'\colon \text{`standard results of } h.s\text{'} \cdot \int_{\overline{s'}} \chi_{post}) \\
= & (\sqcap s'\colon \text{`standard results of } h.s\text{'} \cdot \chi_{post}.s') \\
 & \\
= & \begin{array}{ll} 1 & \text{if `standard results of } h.s\text{'} \subseteq post \\ 0 & \text{otherwise.} \end{array}
\end{array}
$$

In considering only the standard results of $h.s$ (ignoring the closures), we note that the minimum over F of $\int_F \beta$, for any β, is unaffected by leaving out the points generated by closure — up-closure, for example, only increases that value, and convex closure generates only values between the extremes.

Thus $wp.h.\chi_{post}$ is the characteristic function of those initial states all of whose final states produced by h lie in *post*; and that is the standard embedding of $S \to S_\perp$ in $\mathbb{P}S \leftarrow \mathbb{P}S$.

The function *wp* defines a subset of $\mathcal{J}S$, the image of $\mathcal{H}S$ there:

Definition 6.3 The set of *regular probabilistic predicate transformers* $\mathcal{J}_r S$ over S is the *wp*-image of $\mathcal{H}S$ in $\mathcal{J}S$:

$$\mathcal{J}_r S \quad := \quad \{h\colon \mathcal{H}S \cdot wp.h\} \ .$$

□

In the next section we establish properties of $\mathcal{J}_r S$ and in the following section we show that they characterise it, and thus form a probabilistic analogue of conjunctivity.

7 Healthiness conditions for probabilistic programs

In [4] Dijkstra imposes 'healthiness conditions' on the standard predicate transformers $\mathbb{P}S \leftarrow \mathbb{P}S$: they are strictness[11], monotonicity, conjunctivity and continuity. The conditions characterise the images, under the standard embedding, of the total and image-finite relations $[S \leftrightarrow S_\perp]$.

In this section we consider probabilistic healthiness: the probabilistic analogues of strictness, monotonicity, conjunctivity and continuity characterise $\mathcal{J}_r S$.

The analogue of standard conjunctivity is probabilistic sub-linearity:

Definition 7.1 A predicate transformer $j\colon \mathcal{J}S$ is said to be *sub-linear* iff for all $\beta_1, \beta_2\colon \mathcal{P}S$ and $c_0, c_1, c_2\colon \mathbb{R}_{\geq}$

$$c_1(j.\beta_1) + c_2(j.\beta_2) \ominus \boldsymbol{c}_\circ \quad \leq \quad j.(c_1\beta_1 + c_2\beta_2 \ominus \boldsymbol{c}_\circ) \ ,$$

[11]The *Law of the Excluded Miracle.*

where $\ominus$ denotes subtraction in $\mathbb{R}_{\geq}$, so that $x \ominus y = (x - y) \sqcup 0$, and we write $c\beta$ for the pointwise multiplication of the predicate β by the scalar c.

Because $\ominus$ does not associate with $+$, we give it a (lower) syntactic precedence explicitly: by $x + y \ominus z$ we mean $(x + y) \ominus z$. □

We now show that all regular predicate transformers are sub-linear.

Lemma 7.2 *sub-linearity* All members of $\mathcal{J}_r S$ are sub-linear.
Proof: We write $j = wp.h$ for $h: \mathcal{H}S$, and then proceed by direct calculation using the notation of Def. 7.1. For arbitrary $s: S$,

$$
\begin{array}{lll}
 & wp.h.(c_1\beta_1 + c_2\beta_2 \ominus \boldsymbol{c}_\circ).s & (14) \\
= & (\sqcap F: h.s \cdot \int_F (c_1\beta_1 + c_2\beta_2 \ominus \boldsymbol{c}_\circ)) & \text{Def. 6.2} \\
\geq & (\sqcap F: h.s \cdot \int_F (c_1\beta_1 + c_2\beta_2 - \boldsymbol{c}_\circ)) & \text{monotonicity of } \int_F \text{ and } \sqcap \\
= & & \text{distribution of } +, -, \times \text{ through } \int_F \\
 & (\sqcap F: h.s \cdot c_1 \int_F \beta_1 + c_2 \int_F \beta_2 - c_0 \int_F \mathbf{1}) & \\
\geq & (\sqcap F: h.s \cdot c_1 \int_F \beta_1 + c_2 \int_F \beta_2 - c_0) & F \in h.s \subseteq \overline{S} \text{ implies } \int_F \mathbf{1} \leq 1 \\
= & (\sqcap F: h.s \cdot c_1 \int_F \beta_1 + c_2 \int_F \beta_2) - c_0 & \\
\geq & & + \text{ and } (c\times) \text{ monotonic for } c \in \mathbb{R}_{\geq} \\
 & c_1(\sqcap F: h.s \cdot \int_F \beta_1) + c_2(\sqcap F: h.s \cdot \int_F \beta_2) - c_0 & \\
= & c_1(wp.h.\beta_1.s) + c_2(wp.h.\beta_2.s) - c_0 \ , & \text{Def. 6.2}
\end{array}
$$

and since (14) is non-negative we may replace the final $(-c_0)$ by $(\ominus\, c_0)$. □

That sub-linearity is indeed the probabilistic analogue of conjunctivity may be seen by considering a special case: suppose sub-linear $j: \mathcal{J}S$ takes standard postconditions to standard preconditions. We then have

$$
\begin{array}{lll}
 & j.(\chi_{post_1 \cap post_2}) & \\
= & j.(\chi_{post_1} \sqcap \chi_{post_2}) & \\
= & j.(\chi_{post_1} + \chi_{post_2} \ominus \mathbf{1}) & \\
\geq & j.\chi_{post_1} + j.\chi_{post_2} \ominus \mathbf{1} & \text{sub-linearity} \\
= & j.\chi_{post_1} \sqcap j.\chi_{post_2} \ . & j.\chi_{post_{1,2}} \text{ standard}
\end{array}
$$

Equality then follows from monotonicity of j (below).

The other probabilistic healthiness conditions are all consequences of sub-linearity[12]; we treat them in turn, beginning with monotonicity:

[12] They are also easily proved directly from Def. 6.2, a useful exercise.

Lemma 7.3 *monotonicity* If $j{:}\,\mathcal{J}S$ is sub-linear then it is monotonic.

Proof: Take $\beta_1, \beta_2{:}\,\mathcal{P}S$ with $\beta_1 \geq \beta_2$. Then

$$j.\beta_1 \;=\; j.(\beta_1 - \beta_2 + \beta_2) \;\geq\; j.(\beta_1 - \beta_2) + j.\beta_2 \;\geq\; j.\beta_2 \;.$$

$\square$

Probabilistic monotonicity generalises standard monotonicity, just as $\leq$ generalises implication.

Feasibility is expressed as follows:

Lemma 7.4 *feasibility* If $j{:}\,\mathcal{J}S$ is sub-linear, then for all $\beta{:}\,\mathcal{P}S$ we have

$$j.\beta \quad\leq\quad \sqcup\boldsymbol{\beta}\;,$$

where $\sqcup\beta$ abbreviates $(\sqcup s{:}\,S \cdot \beta.s)$.

Proof: We prove first the special case in which $\beta = \mathbf{o}$; for then

$$2 \times j.\mathbf{o} \;\leq\; j.(2 \times \mathbf{o}) \;=\; j.\mathbf{o}\;,$$

so that $j.\mathbf{o} = \mathbf{o}$. Now for the general case we continue

$$\mathbf{o} \;=\; j.\mathbf{o} \;=\; j.(\beta \ominus \sqcup\boldsymbol{\beta}) \;\geq\; j.\beta \ominus \sqcup\boldsymbol{\beta}\;,$$

so that $\sqcup\boldsymbol{\beta} \geq j.\beta$ as required. $\square$

Note that the special case $j.\mathbf{o} = \mathbf{o}$ corresponds to standard strictness, provided we identify $\mathbf{o}$ and *false*.

Another consequence of sub-linearity is the following.

Lemma 7.5 *scaling* If $j{:}\,\mathcal{J}S$ is sub-linear, then for all $\beta{:}\,\mathcal{P}S$ and $c{:}\,\mathbb{R}_{\geq}$

$$j.(c\beta) \quad=\quad c(j.\beta)\;.$$

Proof: Given sub-linearity, we need only prove $\leq$; and for $c \neq 0$ we have

$$j.(c\beta) \;=\; c(1/c)(j.(c\beta)) \;\leq\; c(j.((1/c)c\beta)) \;=\; c(j.\beta)\;.$$

When $c = 0$ the result is immediate from feasibility. $\square$

There seems to be no standard analogue for scaling.

Finally, for continuity we appeal to the finiteness of S.

feasibility	$j.\beta \leq \sqcup\boldsymbol{\beta}$	for $\beta{:}\, \mathcal{P}S$
monotonicity	$\beta_1 \geq \beta_2 \Rightarrow j.\beta_1 \geq j.\beta_2$	for $\beta_1, \beta_2{:}\, \mathcal{P}S$
scaling	$j.(c\beta) = c(j.\beta)$	for $\beta{:}\, \mathcal{P}S$ and $c{:}\, \mathbb{R}_{\geq}$
sub-linearity	$c_1(j.\beta_1) + c_2(j.\beta_2) \ominus \boldsymbol{c}_0$ $\leq\ j.(c_1\beta_1 + c_2\beta_2 \ominus \boldsymbol{c}_0)$	for $\beta_1, \beta_2{:}\, \mathcal{P}S$ and $c_0, c_1, c_2{:}\, \mathbb{R}_{\geq}$
continuity	$j.(\sqcup B) = (\sqcup\beta{:}\, B \cdot j.\beta)$	for $\geq$-directed subset B of $\mathcal{P}S$

Figure 4: Summary of properties characterising $\mathcal{J}_r S$, the regular predicate transformers.[14]

Lemma 7.6 *continuity* If $j{:}\, \mathcal{J}S$ is sub-linear, and B is a $\geq$-directed and bounded subset of $\mathcal{P}S$, then

$$j.(\sqcup B) \quad = \quad (\sqcup\beta{:}\, B \cdot j.\beta)\ .$$

Proof: By monotonicity we need only show $j.(\sqcup B) \leq (\sqcup\beta{:}\, B \cdot j.\beta)$. Since B is directed and bounded, and S is finite, we know that for every $c > 1$ there is a β_c in B large enough that $\sqcup B \leq c\beta_c$. Then we have for any such c

$$\begin{array}{lll} & j.(\sqcup B) & \\ \leq & j.(c\beta_c) & \text{monotonicity} \\ = & c(j.\beta_c) & \text{scaling} \\ \leq & c(\sqcup\beta{:}\, B \cdot j.\beta)\ , & \beta_c \in B \end{array}$$

which suffices since c can be arbitrarily close to 1. □

We summarise the properties of $\mathcal{J}_r S$ in Fig. 4. It remains to show that those properties characterise $\mathcal{J}_r S$ exactly, and for that we define an inverse to *wp*.

[14] In fact sub-linearity is independent of the other properties: let S be the two-element state space $\{a, b\}$, and define $j{:}\, \mathcal{J}S$ so that

$$j.\beta.s \quad := \quad \beta.a + \beta.b - \sqrt{\frac{(\beta.a)^2 + (\beta.b)^2}{2}}\ .$$

Then j is feasible, monotonic, scaling and continuous, and moreover it satisfies *simple sub-linearity*: for all $\beta_1, \beta_2{:}\, \mathcal{P}S$ we have by a routine calculation that

$$j.\beta_1 + j.\beta_2 \quad \leq \quad j.(\beta_1 + \beta_2)\ .$$

But still j is not sub-linear: take $\beta.a, \beta.b := 2, 1$ and consider $j.(\beta \ominus \mathbf{1})$.

8 Characterising regular programs

We now show that sub-linearity characterises the regular predicate transformers $\mathcal{J}_r S$: not only are they sub-linear, but they are the only members of $\mathcal{J}S$ that are.

We begin by defining a map rp from $\mathcal{J}S$ back to $\mathcal{H}S$, to be inverse to wp on $\mathcal{J}_r S$. Thus for $j\colon \mathcal{J}_r S$ in particular, the corresponding relation $rp.j$ should produce from initial s exactly those final distributions F that satisfy (13) for all possible postconditions β:

Definition 8.1 The function $rp\colon \mathcal{J}S \to \mathcal{H}S$ is defined

$$rp.j.s \quad := \quad \{F\colon \overline{S} \mid (\forall \beta\colon \mathcal{P}S \cdot j.\beta.s \le \int_F \beta)\} \ ,$$

for program $j\colon \mathcal{J}S$ and state $s\colon S$. □

Concerning definedness of rp, we note that $rp.j.s$ satisfies the closure conditions for $\mathcal{C}S$: in fact for fixed j, β, s the constraint

$$j.\beta.s \le \int_F \beta$$

on F represents a closed upper half-space in $\mathbb{R}^N$ which satisfies the three closure conditions trivially, and the conditions are preserved by intersection. (It is an upper half-space because β takes only non-negative values.)

If $rp.j.s$ is empty for some s, we consider rp to be undefined at that j; but Lem. 8.5 below shows $rp.j$ to be defined when j is sub-linear.

Our first use of rp is to show that wp is indeed an injection:

Lemma 8.2 For any $h\colon \mathcal{H}S$ we have

$$rp.(wp.h) \quad = \quad h \ .$$

Proof: Direct from the definitions we have (taking the contrapositive), for arbitrary $F\colon \overline{S}$,

	$F \notin rp.(wp.h).s$	
iff	$F \notin \{F\colon \overline{S} \mid (\forall \beta\colon \mathcal{P}S \cdot wp.h.\beta.s \le \int_F \beta)\}$	Def. 8.1
iff	$(\exists \beta\colon \mathcal{P}S \cdot wp.h.\beta.s > \int_F \beta)$	
iff	$(\exists \beta\colon \mathcal{P}S \cdot (\sqcap F'\colon h.s \cdot \int_{F'} \beta) > \int_F \beta)$	Def. 8.1
iff	$(\exists \beta\colon \mathcal{P}S; r\colon \mathbb{R} \cdot (\forall F'\colon h.s \cdot \int_{F'} \beta > r) \wedge r > \int_F \beta)$	
iff	$F \notin h.s$.	see below

The deferred 'only if' is trivial. For 'if' we need the *separating hyperplane lemma*, which states that any point not in a closed convex subset of $\mathbb{R}^N$ can be separated from it by a hyperplane. (See App. A.)

Now $h.s$ is closed and convex, by definition of $\mathcal{H}S$, and the inequalities $\int_{F'} \beta > r$ and $r > \int_F \beta$ would place all F''s in $h.s$ one one side, and F on the other side, of a hyperplane in $\mathbb{R}^N$ determined by β and r. Thus Lem. A.1 gives us that hyperplane, and we have our β and r once we notice that up-closure of $h.s$ guarantees that β — which is the vector normal to the plane — takes only non-negative values. □

The Cauchy-closure condition on $h.s$ is essential for Lem. 8.2: if $h.s$ were for example

$$\{p\colon (0,1) \cdot \overline{0}\ {}_p\!\oplus \overline{1}\}\ ,$$

thus all probabilitistic combinations of the point distributions $\overline{0}$ and $\overline{1}$ excluding the endpoints, then $rp.(wp.h).s$ would be

$$\{p\colon [0,1] \cdot \overline{0}\ {}_p\!\oplus \overline{1}\}\ ,$$

in which closure has occurred.

Our next step is to prove an analogous result for the opposite direction; but since rp is not defined for all $\mathcal{J}S$ (for $j \notin \mathcal{J}_r S$ it may be that $rp.j.s$ is empty for some s), we state the lemmas conditionally.

Our first lemma concerns members of $\mathcal{J}S$ generally:

Lemma 8.3 For any $j\colon \mathcal{J}S$, if $rp.j$ is defined then

$$wp.(rp.j) \quad \sqsupseteq \quad j.$$

Proof: Take any $\beta\colon \mathcal{P}S$ and $s\colon S$; directly from the definitions we have:

	$wp.(rp.j).\beta.s \geq j.\beta.s$	
iff	$(\sqcap F\colon rp.j.s \cdot \int_F \beta) \geq j.\beta.s$	Def. 6.2; $rp.j$ defined by assumption
iff	$(\forall F\colon rp.j.s \cdot \int_F \beta \geq j.\beta.s)$	
iff	$(\forall F\colon \overline{S} \cdot (\forall \beta'\colon \mathcal{P}S \cdot j.\beta'.s \leq \int_F \beta') \Rightarrow \int_F \beta \geq j.\beta.s)$	
iff	$true$.	

□

Our second lemma is restricted to sub-linear elements of $\mathcal{J}S$:

Lemma 8.4 If $j\colon \mathcal{J}S$ is sub-linear and $rp.j$ is defined, then

$$wp.(rp.j) \quad \sqsubseteq \quad j\ .$$

Proof: We proceed by contradiction: for arbitrary $s\colon S$ and $\beta\colon \mathcal{P}S$,

$$
\begin{array}{ll}
 & wp.(rp.j).\beta.s > j.\beta.s \\
\text{iff} & (\sqcap F\colon rp.j.s \cdot \textstyle\int_F \beta) > j.\beta.s \quad \text{Def. 6.2; } rp.j \text{ defined by assumption} \\
\text{implies} & (\forall F\colon rp.j.s \cdot \textstyle\int_F \beta > j.\beta.s) \hfill (15) \\
\text{iff} & (\forall F\colon \overline{S} \cdot (\forall \beta'\colon \mathcal{P}S \cdot j.\beta'.s \le \textstyle\int_F \beta') \Rightarrow \textstyle\int_F \beta > j.\beta.s) \qquad \text{Def. 8.1} \\
\text{iff} & (\cap \beta'\colon \mathcal{P}S \cdot \{F\colon \overline{S} \mid j.\beta'.s \le \textstyle\int_F \beta'\}) \subseteq \{F\colon \overline{S} \mid \textstyle\int_F \beta > j.\beta.s\} \\
 & \\
\text{iff} & \hfill A \subseteq B \text{ iff } A \cap B^C = \emptyset \\
 & \qquad\qquad (\cap \beta'\colon \mathcal{P}S \cdot \{F\colon \mathbb{R}^N \mid j.\beta'.s \le \textstyle\int_F \beta'\}) \hfill (16) \\
 & \qquad \cap \quad \{F\colon \mathbb{R}^N \mid -j.\beta.s \le \textstyle\int_F (-\beta)\} \\
 & \qquad \cap \quad \{F\colon \mathbb{R}^N \mid -1 \le \textstyle\int_F (-\mathbf{1})\} \\
 & = \ \emptyset\ ,
\end{array}
$$

where we have negated the second term to make the inequalities uniform. The third term, also negated, is added because of the re-typing of the bound variables F: in (16) they are taken from all of $\mathbb{R}^N$, rather than the more restrictive $\overline{S}$, and so we must compensate by requiring explicitly

$$\sum F \quad = \quad \int_F \mathbf{1} \quad \le \quad 1\ .$$

(The remaining constraints $F.s \ge 0$ applying to members F of $\overline{S}$, for every $s\colon S$, are included already in the first term above: take $\beta' := \chi_{\{s\}}$ and recall that $0 \le j.\beta'.s$.)

Now we argue that some finite sub-collection of the sets (16) has empty intersection also. Take the sets determined by $\beta' := \chi_{\{s\}}$ for each $s\colon S$ (finitely many), and the final set $\{F\colon \mathbb{R}^N \mid -1 \le \int_F -\mathbf{1}\}$: they determine a closed hyper-pyramid in the positive hyper-octant of $\mathbb{R}_N$ (whose apex points downwards towards the origin). Since the pyramid is bounded, it is compact — and so the finite intersection lemma[15] applies within it. Thus we can restrict our attention to a collection of just M, say, of the sets (16).

[15]The *finite intersection lemma* states that a collection of closed subsets of a compact set has non-empty intersection if all finite sub-collections of it do.

We now appeal to another lemma in the style of linear programming (App. A). The finite M-collection from (16) with empty intersection may be regarded as a system of M equations

$$A \cdot x \quad \geq \quad r \tag{17}$$

that has no solution in x, where A is an $M \times N$ matrix (of coefficients, representing the β''s, $-\beta$ and $-\mathbf{1}$), x is an $N \times 1$ column vector (representing points F in $\mathbb{R}^N$), and r is an $M \times 1$ column vector (of constant terms, representing the $j.\beta'.s$'s, $-j.\beta.s$ and -1). The expression $A \cdot x$ denotes matrix multiplication, and the inequality $\geq$ is taken row-wise.

Lem. A.2 then gives us a $1 \times M$ row-vector C of non-negative reals such that

$$C \cdot A = 0 \quad \text{but} \quad C \cdot r > 0 \ .$$

Returning to (16), we thus have a finite number of non-negative coefficients $c, c_0 \cdots c_{M-2}$ such that

$$\begin{array}{rcl} c\beta & = & c_1\beta'_1 + \cdots + c_{M-2}\beta'_{M-2} - \boldsymbol{c}_\circ \\ \text{but} \quad c(j.\beta.s) & < & c_1(j.\beta'_1.s) + \cdots c_{M-2}(j.\beta'_{M-2}.s) - c_0 \ . \end{array}$$

We then finish with

$$\begin{array}{lll} & c(j.\beta.s) & \\ < & c_1(j.\beta'_1.s) + \cdots + c_{M-2}(j.\beta'_{M-2}.s) \ominus c_0 & \text{above} \\ \leq & j.(c_1\beta'_1 + \cdots + c_{M-2}\beta'_{M-2} \ominus \boldsymbol{c}_\circ).s \ , & \text{sub-linearity of } j \\ = & j.(c\beta).s \ , & \text{above} \end{array}$$

which contradicts scaling (implied by sub-linearity). □

Finally we deal with definedness of rp:

Lemma 8.5 If $j{:}\,\mathcal{J}S$ is sub-linear, then $rp.j$ is defined.
Proof: The three closure conditions have already been dealt with. For non-emptiness, again we proceed by contradiction: suppose for some $s{:}\,S$ that $rp.j.s$ is empty. Then we have immediately

$$(\forall F{:}\, rp.j.s \cdot \int_F \beta > j.\beta.s) \ ,$$

by quantifying universally over the empty set (the body of the quantification is irrelevant). But that is identical to (15) in the proof of Lem. 8.4, and rp is not mentioned beyond that point. Thus we proceed to a contradiction as before. □

We thus have thus established the following:

Lemma 8.6 If $j{:}\,\mathcal{J}S$ is sub-linear, then $rp.j$ is defined and

$$wp.(rp.j) \quad = \quad j \ .$$

Proof: The result is immediate from Lemmas 8.3–8.5. □

With Lemmas 7.2 and 8.6 we have finally our characterisation of $\mathcal{J}_r S$:

Theorem 8.7 The regular predicate transformers $\mathcal{J}_r S$ are characterised by sub-linearity: j in $\mathcal{J}S$ is sub-linear iff it is in $\mathcal{J}_r S$.
Proof: 'If' is given by Lem. 7.2; 'only if' is given by Lem. 8.6. □

Note that Thm. 8.7 gives also the conditions of Fig. 3 as characteristic collectively of $\mathcal{J}_r S$, since Lemmas 7.2–7.6 show they are all implied by sub-linearity.

9 Non-determinism and predicate transformers

We have now seen that $\mathcal{H}S$ and $\mathcal{J}_r S$ are placed in 1-1 correspondence by the mutual inverses $wp{:}\,\mathcal{H}S \to \mathcal{J}_r S$ and $rp{:}\,\mathcal{J}_r S \to \mathcal{H}S$. But for the two spaces to give equivalent semantics for regular programs requires further that the program operators preserve the correspondence — that for example Definitions 5.5–5.7, for relations, correspond with the definitions of Fig. 2 for predicate transformers.

In fact [11] does not treat non-determinism, and thus in the case of Def. 5.6 rather than prove a correspondence we must generate a definition: for $h_1, h_2{:}\,\mathcal{H}S$, $\beta{:}\,\mathcal{P}S$ and $s{:}\,S$ we calculate

$$
\begin{array}{lll}
 & wp.(h_1 \sqcap h_2).\beta.s & \\
= & (\sqcap F{:}\,(h_1 \sqcap h_2).s \cdot \textstyle\int_F \beta) & \text{Def. 6.2} \\
= & (\sqcap F_1{:}\,h_1.s;\, F_2{:}\,h_2.s;\, p{:}\,[0,1] \cdot \textstyle\int_{F_1\,{}_p\!\oplus F_2} \beta) & \text{Def. 5.6} \\
= & (\sqcap F_1{:}\,h_1.s;\, F_2{:}\,h_2.s;\, p{:}\,[0,1] \cdot p(\textstyle\int_{F_1} \beta) + (1-p)(\textstyle\int_{F_2} \beta)) & \text{Def. 3.1} \\
 & & \\
= & \qquad\qquad \text{minimum occurs at extremes, thus at } p = 0 \text{ or } p = 1 & \\
 & (\sqcap F_1{:}\,h_1.s;\, F_2{:}\,h_2.s \cdot \textstyle\int_{F_1} \beta \sqcap \textstyle\int_{F_2} \beta) &
\end{array}
$$

$$
\begin{array}{ll}
= & (\sqcap F_1 \colon h_1.s \cdot \int_{F_1} \beta) \sqcap (\sqcap F_2 \colon h_2.s \cdot \int_{F_2} \beta) \\
= & wp.h_1.\beta.s \sqcap wp.h_2.\beta.s\ . \qquad \text{Def. 6.2}
\end{array}
$$

Thus we are given a definition of non-determinism over $\mathcal{J}_r S$, which we extend to $\mathcal{J} S$:

Definition 9.1 For $j_1, j_2 \colon \mathcal{J} S$, $\beta \colon S$ and $s \colon S$,

$$(j_1 \sqcap j_2).\beta.s \quad := \quad j_1.\beta.s \sqcap j_2.\beta.s\ .$$

□

Variations on example (11) from Sec. 4 illustrate well the interaction of probabilistic choice and demonic non-determinism. Suppose first that x is chosen non-deterministically (rather than probabilistically), and that we take postcondition $x = y$. Then

$$
\begin{array}{ll}
 & wp.(x := 1 \sqcap x := 2;\ y := 1\ {}_{\frac{1}{2}}\!\oplus y := 2).(x = y) \\
= & wp.(x := 1 \sqcap x := 2).((x = 1)/2 + (x = 2)/2) \\
= & ((1 = 1)/2 + (1 = 2)/2) \sqcap ((2 = 1)/2 + (2 = 2)/2) \\
= & (1/2 + 0/2) \sqcap (0/2 + 1/2) \\
= & 1/2\ ,
\end{array}
$$

from which we see that program establishes $x = y$ with probability at least $1/2$: no matter which value is assigned to x, with probability $1/2$ the second command will assign the same to y.

Now suppose instead that it is the second choice that is non-deterministic. Then we have

$$
\begin{array}{ll}
 & wp.(x := 1\ {}_{\frac{1}{2}}\!\oplus x := 2;\ y := 1 \sqcap y := 2).(x = y) \\
= & wp.(x := 1\ {}_{\frac{1}{2}}\!\oplus x := 2).((x = 1) \sqcap (x = 2)) \\
= & ((1 = 1) \sqcap (1 = 2))/2 + ((2 = 1) \sqcap (2 = 2))/2 \\
= & (1 \sqcap 0)/2 + (0 \sqcap 1)/2 \\
= & 0\ ,
\end{array}
$$

reflecting that no matter what value is assigned probabilistically to x, the demon could choose subsequently to assign a different value to y.

Thus it is clear that the execution order of occurrence of the two choices plays a critical role in their interaction, and in particular that the demon in the first case cannot make the assignment 'clairvoyantly' to x in order to avoid the value that later will be assigned to y.

It is straightforward, with calculations like that preceeding Def. 9.1, to establish the correspondence of the remaining relational operators with their predicate-transformer counterparts. Usually the latter are simpler.

10 Example: termination and recursion

Def. 8.1 suggests that a program $j\colon \mathcal{J}_r S$ should be said to terminate from initial state $s\colon S$ just when $j.\mathbf{1}.s = 1$; for then we have $1 \leq \int_F \mathbf{1}$ for the final distributions F that $rp.j$ delivers, and the discussion preceding Def. 2.2 characterises those as terminating. By monotonicity and scaling (both properties of $\mathcal{J}_r S$), that is equivalent to the following definition:

Definition 10.1 *termination* A program $j\colon \mathcal{J}_r S$ is said to *terminate* at an (initial) state $s\colon S$ if for all $\beta\colon \mathcal{P} S$ we have

$$\sqcap \beta \quad \leq \quad j.\beta.s\ .$$

$\square$

We have already a similar characterisation of feasibility: at a state s, Lem. 7.4 requires $j.\beta.s \leq \sqcup \beta$, so that we have a duality between the two notions just as in the standard case;[16] and j is thus both feasible and terminating (at all states) just when for all $\beta\colon \mathcal{P} S$

$$\sqcap \boldsymbol{\beta} \quad \leq \quad j.\beta \quad \leq \quad \sqcup \boldsymbol{\beta}\ .$$

We now consider an example of probabilistic termination, the recursive program

$$G \quad := \quad \mathbf{skip}\ {}_{\frac{1}{2}}{\oplus}\ G \tag{18}$$

that chooses repeatedly with probability 1/2 between termination and (fruitless) recursion. From Fig. 2 we have, for any $\beta\colon \mathcal{P} S$,

$$wp.G.\beta \quad = \quad \beta/2 + (wp.G.\beta)/2$$

[16] The standard duality is that termination is indicated by $true \Rightarrow j.true$, and feasibility by $j.false \Rightarrow false$.

whose unique solution is given by $wp.G.\beta = \beta$. Thus by Def. 10.1 we see that G terminates at every state; indeed, from Fig. 2 we have that $G = \textbf{skip}$.

The above reasoning goes through for any ${}_p\oplus$ when $0 < p$, and illustrates well how the familiar concept of 'termination with probability 1' appears for us here: the program G has probability $(1-p)^n$ of recursing at least n times, and thus for non-zero p has probability 0 of recursing forever (as $n \to \infty$).

Treatment of recursions in general, however, relies on the existence of fixed points in $\mathcal{J}_r S$. If we consider only feasible programs, then it is easy to see that $\sqsubseteq$-directed sets of programs have least upper bounds, in spite of the fact that limits in $\mathcal{P}S$ do not necessarily exist: for any directed set $\mathcal{G}$ of programs, its limit is defined pointwise — thus for any $\beta\colon \mathcal{P}S$

$$(\sqcup\mathcal{G}).\beta \quad := \quad (\sqcup G\colon \mathcal{G} \cdot G.\beta)\ ,$$

and the limit exists on the right because feasibility gives $G.\beta \leq \sqcup\beta$ for all elements G of $\mathcal{G}$. Since sub-linearity is preserved by taking limits, by continuity of the arithmetic operators, we have $\sqcup\mathcal{G} \in \mathcal{J}_r S$ provided $\mathcal{G} \subseteq \mathcal{J}_r S$.

Thus with continuity, recursions are given meaning via the usual ω-limit construction. Consider for example the program over $\mathbb{N}$[17]

$$\begin{array}{l} n\colon= \ 0; \\ (\mu G \cdot \textbf{skip} \ {}_{\frac{1}{2}}\oplus \ n\colon= \ n+1; G)\ . \end{array} \tag{19}$$

Beginning with its second statement, we calculate $wp.G.(n = N)$ as the limit $(\sqcup i\colon \mathbb{N} \cdot \gamma_i)$, where

$$\begin{array}{rcll} \gamma_0 & = & \textbf{abort}.(n = N) & = 0 \\ \gamma_{i+1} & = & (n = N)/2 + (\gamma_i[n\colon= n+1])/2\ . & \end{array}$$

Induction shows that $\gamma_i = \sum_{i'<i}(n + i' = N)/2^{i'+1}$, and thus taking the limit we find that

$$wp.G.(n = N) \quad = \quad \sum_i (n + i = N)/2^{i+1}\ .$$

Hence for the whole program we have

[17]This example comes from [11], and so we relax our assumption that S is finite in order to make our later point about unbounded non-determinism.

$$
\begin{array}{lll}
& wp.(n\!:=\ 0; G).(n = N) & \\
= & wp.(n\!:=\ 0).(\sum_i (n+i = N)/2^{i+1}) & \text{above} \\
= & \sum_i (i = N)/2^{i+1} & \text{Fig. 2 for } n\!:=\ 0 \\
= & 1/2^{N+1}\ , &
\end{array}
$$

showing that Program (19) sets n finally to N with probability $1/2^{N+1}$. That it terminates can be seen as in our earlier example; alternatively, sub-linearity and continuity can be used to sum the probabilities for all N, giving (at least) 1 as a total.

Note that we have found an 'unboundedly probabilistic' program — every one of its infinitely many final states is accessible, however small the associated probability might be. But that differs from unbounded non-determinism, and so the program (as a predicate transformer) remains continuous. Indeed Program (19) contains no non-determinism at all: it is maximally feasible, hence deterministic.

11 Comparisons and conclusions

The key to a systematic exploration of probabilistic predicate transformers seems to have been the fortuitous correspondence of Jones' work with the more recent work of He. The latter we modified slightly — using sub-probability measures (not summing to 1) to avoid an explicit bottom state $\perp$, and imposing Cauchy closure on the result sets. Dijkstra's presentation of standard healthiness conditions, and their dependence on the embedding of standard relations in the standard predicate transformers, then suggested the approach reported here.

Rao extends UNITY [3] with a statement that chooses between its components with some unknown but non-zero probability for each one [26, 25]. Restrictions on the execution of the program as a whole ('unconditional', and 'extreme' fairness) then allow conclusions to be drawn about the eventual execution of each probablistic alternative, independent of the actual probabilities concerned; and from that an extension of the UNITY logic allows properties (for example termination) to be proved to hold with probability 1.

Given components j and j', the program

$$j \;{}_p\!\oplus\; j' \quad \sqcap \quad j' \;{}_{p'}\!\oplus\; j$$

executes j, j' with probability at least p, p' respectively, provided $p + p' \leq 1$; but our Cauchy closure (equivalently continuity, or bounded non-determinism) prevents us from abstracting from the specific values of p, p': we cannot specify that they are arbitrarily small but non-zero.

Rao's non-determinism is less demonic than ours; with the unconditional fairness assumption of UNITY (applied by analogy to an explicit recursion) the program

$$(\mu G \cdot \mathbf{skip} \sqcap G)$$

would terminate because the body cannot be executed indefinitely without selecting its first component **skip**. However under the standard interpretation [4], thus ours as well, that program is equivalent to **abort**.

To match the elegance of Rao's termination proofs (for example [26, Sec. 15]) we may need theorems for the probabilistic analogue of variants and invariants; although with explicit probabilities we can express more, we pay the price of operating at a lower level. Probabilistic variants in particular may come to be based directly on the 'random walk' or 'gambler's ruin' [7], rather than on the simpler 'counting down to zero' paradigm that suffices for standard programs. For example, it can be shown that the program

$$\begin{array}{l} \mathbf{do}\ x \neq 0 \rightarrow \\ \quad x{:=}\ x+1 \ {}_{\frac{1}{2}}\oplus\ x{:=}\ x-1 \\ \mathbf{od}\ , \end{array}$$

where $x{:}\,\mathbb{Z}$, terminates with probability 1. (An approach as for Program (18) suffices.) It exhibits what could be a 'symmetric random walk style' variant (simply x in this case), which pattern could be exploited for total correctness of more general iterations. But note that Program (18) itself terminates in spite of having no variables over which a variant could be formed; thus in some cases they may need to be added.

There is an abundance of other work on probability and concurrency; but (to our knowledge) there has been no exploration of probabilistic predicate transformers except for Rao, and Jones and Plotkin themselves.

Other investigations of the link between non-determinism and probability address the situation of probabilistic programs running concurrently under a non-deterministic scheduler [24, 8, 14], often in the framework of temporal logic. It is shown that the 'extent' of non-determinism in the scheduler is crucial: can it take advantage of probabilistic choices already made, or even yet to be made? Here, like [14], our non-determinism is demonic rather than indifferent (it takes advantage of earlier probabilistic choices), but not clairvoyant (it cannot take advantage of probabilistic choices not yet made).

There is much more to say about probabilistic predicate transformers:[18]

1. The results concerning the functions wp and rp, and the healthiness conditions, extend to infinite state spaces. There we find that the key to continuity in $\mathcal{J}_r S$ is the Cauchy-closure in $\mathcal{H}S$, which excludes unbounded non-determinism.

2. Deterministic programs, at least in $\mathcal{J}_r S$, are characterised by (exact) linearity, so that j is deterministic whenever for all β_1, β_2 and c_1, c_2 we have

$$c_1(j.\beta_1) + c_2(j.\beta_2) \quad = \quad j.(c_1\beta_1 + c_2\beta_2)\ .$$

   That can be shown to specialise to disjunctivity in the standard case.

3. There is a Galois connection between the standard and the probabilistic predicate transformers, embedding the former in the latter, and monotonic standard programs j are characterised in the probabilistic space by the equality

$$c(j.\beta) \ominus \boldsymbol{c}_\circ \quad = \quad j.(c\beta - \boldsymbol{c}_\circ)\ .$$

   The embedding is more general than that proposed by He [9], because we consider not only relational programs.

   Mapping probabilistic programs back into the standard space in many cases simply replaces probabilistic choice by non-deterministic choice.

4. If the standard healthiness conditions are relaxed, dropping in particular feasibility (strictness) [23, 17, 22] and conjunctivity [2, 6], then every standard monotonic predicate transformer can be expressed as the least upper bound of conjunctive standard predicate transformers [2].

[18] Because of lack of space we defer full treatments to a later report [20].

The same is true in the probabilistic space: replacing sub-linearity by the weaker

$$c_1(j.\beta) \ominus \boldsymbol{c}_0 \quad \leq \quad j.(c_1\beta \ominus \boldsymbol{c}_0) \ ,$$

for all $\beta\colon \mathcal{P}S$ and $c_0, c_1\colon \mathbb{R}_{\geq}$, characterises a superset of $\mathcal{J}_r S$ large enough to include all standard monotonic predicate transformers (not just the relational ones). Analogously to the standard case, every element of that larger subset of $\mathcal{J}S$ is expressible as the least upper bound of sub-linear predicate transformers.

The functions *wp* and *rp* then form a Galois connection between the relational and predicate-transformer models.

A possible limitation of the simple definition of distribution employed is its inability to express even the uniform distribution over $[0, 1]$. That does not affect our ability to reason about programs whose only probabilistic operator is ${}_p\oplus$, however, since with only that (and even with recursion) such an 'infinitarily-unbiased' program cannot be written. But still one might wish to analyse the behaviour of a given program over an externally-determined distribution of inputs, and the limitation may operate at that point.

Still further work is to consider probabilistic 'specification statements' [1, 22, 17, 18], for example of the form

$$w : [\alpha, \beta]$$

for list of program variables w and probabilistic predicates α, β. That may lead to a probabilistic refinement calculus, although initial investigations suggest that the probabilistic specification statements are (relatively) less expressive than their standard counterparts.

Finally, we are not the first to notice the connection between probabililty and linear programming (for example [5]).

Acknowledgements

We thank Jifeng He and Tony Hoare for their comments and contributions.

References

[1] R.-J.R. Back. A calculus of refinements for program derivations. *Acta Informatica*, 25:593–624, 1988.

[2] R.-J.R. Back and J. von Wright. Duality in specification languages: a lattice theoretical approach. *Acta Informatica*, 27:583–625, 1990.

[3] K.M. Chandy and J. Misra. *Parallel Program Design: A Foundation*. Addison-Wesley, 1988.

[4] E.W. Dijkstra. *A Discipline of Programming*. Prentice Hall International, Englewood Cliffs, 1976.

[5] R. Fagin, J.Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Information and Computation*, 87:78–128, 1990.

[6] P.H.B. Gardiner and C.C. Morgan. Data refinement of predicate transformers. *Theoretical Computer Science*, 87:143–162, 1991. Reprinted in [21].

[7] G. Grimmett and D. Welsh. *Probability: an Introduction*. Oxford Science Publications, 1986.

[8] S. Hart, M. Sharir, and A. Pnueli. Termination of probabilistic concurrent programs. *ACM Transactions on Programming Languages and Systems*, 5:356–380, 1983.

[9] Jifeng He, A.K. McIver, and K. Seidel. Probabilistic models for the guarded command language. In preparation, Programming Research Group, Oxford, 1995.

[10] C.A.R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, 583, October 1969.

[11] C. Jones. *Probabilistic Nondeterminism*. PhD thesis, Edinburgh University, 1990. Available as Technical Report ECS-LFCS-90-105.

[12] C. Jones and G. Plotkin. A probabilistic powerdomain of evaluations. In *Proceedings 4th IEEE Annual Symposium on Logic in Computer Science*, pages 186–195, Cambridge, Mass., March 1989. IEEE Computer Society.

[13] D. Kozen. Semantics of probabilistic programs. *Journal of Computer and System Sciences*, 22:328–350, 1981.

[14] D. Lehmann and M.O. Rabin. On the advantages of free choice: a symmetric and fully-distributed solution to the Dining Philosophers Problem. In *Proceedings of the 8th Annual ACM Symposium on Principles of Programming Languages*, pages 133–138. ACM, 1981.

[15] G. Lowe. Representing nondeterministic and probabilistic behaviour in reactive processes. Programming Research Group, Oxford.

[16] Carroll Morgan, Annabelle McIver, Karen Seidel, and JW Sanders. Probabilistic predicate transformers. Technical Report PRG-TR-4-95, Programming Research Group, February 1995.

[17] C.C. Morgan. The specification statement. *ACM Transactions on Programming Languages and Systems*, 10(3), July 1988. Reprinted in [21].

[18] C.C. Morgan. *Programming from Specifications.* Prentice-Hall, second edition, 1994.

[19] C.C. Morgan, A.K. McIver, K. Seidel, and J.W. Sanders. Refinement-oriented probability for CSP. Technical Report PRG-TR-12-94, Programming Research Group, August 1994.

[20] C.C. Morgan, A.K. McIver, K. Seidel, and J.W. Sanders. Further properties of probabilistic predicate transformers. In preparation, Programming Research Group, Oxford, 1995.

[21] C.C. Morgan and T.N. Vickers, editors. *On the Refinement Calculus.* FACIT Series in Computer Science. Springer Verlag, 1994.

[22] J.M. Morris. A theoretical basis for stepwise refinement and the programming calculus. *Science of Computer Programming*, 9(3):287–306, December 1987.

[23] G. Nelson. A generalization of Dijkstra's calculus. *ACM Transactions on Programming Languages and Systems*, 11(4):517–561, October 1989.

[24] M.O. Rabin. The choice-coordination problem. *Acta Informatica*, 17(2):121–134, June 1982.

[25] J.R. Rao. *Building on the UNITY Experience: Compositionality, Fairness and Probability in Parallelism.* PhD thesis, University of Texas at Austin, 1992.

[26] J.R. Rao. Reasoning about probabilistic parallel programs. *ACM Transactions on Programming Languages and Systems*, 16(3), May 1994.

[27] A. Schrijver. *Theory of Integer and Linear Programming.* Wiley, New York, 1986.

[28] K. Trustrum. *Linear Programming.* Library of Mathematics. Routledge and Kegan Paul, 1971.