PROBABILISTIC COMMUNICATING PROCESSES

by

Karen Seidel

Technical Monograph PRG-102
ISBN 0-902928-79-1

Michaelmas 1992

Oxford University Computing Laboratory
Programming Research Group
11 Keble Road
Oxford OX1 3QD
England

Oxford University Compnting Laboratory
Programming Research Group
11 Keble Road
Oxford OX1 3QD
England

Probabilistic Communicating Processes

Karen Seidel

Lady Margaret Hall

Thesis submitted for the degree of Doctor of Philosophy
at the University of Oxford

Hilary Term, 1992

Abstract

In this thesis, we develop a mathematical formalism for the specification and proof of correctness of probabilistic communicating processes. This formalism combines a notion of probabilistic correctness with the theory of concurrency provided by the language of Communicating Sequential Processes (CSP).

We first present the semantics of a model in which processes are defined as probability measures on the space of infinite traces. The model contains definitions for prefixing, probabilistic choice, hiding, simple parallel composition, sequential composition, interleaving, relabelling and recursion. These operators are defined as functions (mostly transformations) of probability measures. Although the semantics of this model is very different from that of other models of CSP, it has almost the same algebraic properties as the traces model. Examples are given which use these algebraic properties as well as the probabilistic properties of the processes.

In the second part of the thesis we present the semantics of a model in which processes are defined as conditional probability measures. This enables us to give definitions for external choice and alphabetised parallel composition, as well as prefixing, probabilistic choice, relabelling and recursion. Again we show that this semantics satisfies the appropriate algebraic laws. We also present a set of proof rules which provide a link between the process algebra and behavioural specifications. A significant case study is used to demonstrate the applicability of the model and the proof rules.

Acknowledgements

Above all, I would like to thank my supervisor, Jeff Sanders, for his continuing guidance and advice as well as his friendship. I would also like to acknowledge the many helpful comments made by Steve Schneider, Bill Roscoe, Isaac Saias, Jeremy Jacob and Tony Hoare. I would further like to thank Jim Davies, without whom the typesetting of this thesis would have been quite scrappy.

Finally I would like to give a special mention to Clare Martin, my comrade in arms and moral support extraordinaire.

This research was made possible by a grant from the Science and Engineering Research Council.

Dedication

Für meine Eltern

Contents

Chapter 1

Introduction

Randomised algorithms are increasingly being used in distributed systems, for instance to solve problems like load-balancing [Pu90] and self-stabilisation [Herm90]. Not only are these algorithms often simpler and faster than any deterministic alternative but sometimes no such alternative exists, as in the case of randomised consensus [AH90]. This is typically so when identical components in identical situations have to make different decisions for the system to progress.

A mathematical formalism for the specification of systems involving randomisation must be based on a notion of probabilistic correctness. For a deterministic algorithm the statement that it is correct in that it has or achieves a certain property is either true of false. Moreover, if the property is to be achieved, this is guaranteed to happen within a finite number of steps. By contrast, a randomised algorithm is correct if it has or achieves a property with probability 1. So there may be possible behaviours of the algorithm which violate the property in question; only the probability that they will happen is 0. Also, if the property is to be achieved, we cannot give a finite bound on when this will happen, only that it will be within a finite expected number of steps.

The language of Communicating Sequential Processes (*CSP*) [Hoa85] provides a mathematical formalism for the specification of deterministic distributed systems. Its main advantages are support for algebraic reasoning and an effective treatment of concurrency. Our aim is to construct a probabilistic version of *CSP* which combines these advantages with a notion of probabilistic correctness.

Chapter 2 of this thesis contains some measure theory which we will need to formalise this notion. Chapter 3 presents a small model which differs from standard *CSP* in that it has a probabilistic choice operator instead of the internal choice operator of *CSP*, no external choice, and a parallel composition operator only for processes which synchronise on every action. Processes are defined as probability measures on the space of infinite traces and operators as functions (mostly transfor-

mations) of probability measures. We chose to work with infinite rather than finite traces because many probabilistic considerations are about asymptotic behaviours and thus involve taking limits to infinity and cannot be expressed in terms of finite traces. Also by defining all probabilities on one infinite-dimensional space, rather than on many different finite-dimensional spaces, we can use standard concepts of convergence of probability measures which are crucial for the definition of recursion, as presented in chapter 4. In chapter 5 we give examples to show how to prove that probabilistic processes have the properties that distinguish them from deterministic processes.

Chapters 6 and 7 extend the model to include alphabetised parallel composition and external choice. The model of chapter 7 defines processes as conditional probability measures. This is motivated by our understanding of external choice in a probabilistic context: given that the environment has chosen a certain action, a process offering external choice will engage in this action with probability 1. Thus external choice is most naturally defined as a conditional probability and all the definitions given in chapter 3 can be modified to apply to conditional probabilities, too. Chapter 8 contains a set of proof rules based on the model in chapter 7. We use these rules as well as the theory developed in the earlier chapters in chapter 9 where we present the formal specification and proof of correctness of a randomised consensus protocol. The algorithm is a variation of an algorithm devised by Aspnes and Herlihy [AH90]. Our version is guaranteed to terminate only if the scheduling of the components of the protocol is independent of the state of the components, but the expected number of steps to termination is $\mathcal{O}(n^2)$, as opposed to $\mathcal{O}(2^n)$ for the original algorithm, and $\mathcal{O}(n^4)$ for the best previously known algorithm.

Chapter 2

Preliminary Material

2.1 Probability Theory

This section contains definitions and results from probability theory which we will need later. Proofs of the results can be found in Billingsley[Bi79] or Shiryayev[Sh84] (or indeed in any good textbook on probability theory).

Let Ω be a set of points. A σ-field $\mathcal{F}$ defined on Ω is a family of sets on Ω which contains Ω and is closed under the formation of complements as well as finite and countable unions. (A *field* is closed only under complementation and finite unions.) Members of a σ-field are called *measurable* sets. The pair $(\Omega, \mathcal{F})$ constitutes a *measurable space*. A *probability space* consists of a measurable space $(\Omega, \mathcal{F})$ and a probability measure P defined on $\mathcal{F}$. A *probability measure* P on a field or σ-field $\mathcal{F}$ is a function $P : \mathcal{F} \to \mathbb{R}$ which satisfies the following conditions:

1. $\forall A \in \mathcal{F} \cdot \; 0 \leq P A \leq 1$,
2. $P \emptyset = 0, \;\; P \Omega = 1$,
3. if $(A_n)_{n:\mathbb{N}}$ forms a disjoint sequence of $\mathcal{F}$-sets (and, if $\mathcal{F}$ is only a field, $\bigcup_n A_n \in \mathcal{F}$) then

$$P \bigcup_n A_n \;=\; \sum_n P A_n .$$

The last condition is called *countable additivity*. If a function $P : \mathcal{F} \to \mathbb{R}$ satisfies conditions 1. and 2. and is finitely additive then it can be shown that it is countably additive if and only if it is continuous in the sense that

$$A_n \downarrow A \;\; \Rightarrow \;\; P A_n \downarrow P A$$

meaning if $(A_n)_{n:\mathbb{N}}$ is a sequence of sets such that $A_{n+1} \subset A_n$ and $A = \bigcap_n A_n$ then the probabilities $P A_n$ approach $P A$ from above.

To reason about a σ-field it is sometimes useful to know that it is a *monotonic class*, which is a collection $\mathcal{M}$ of subsets of Ω such that if $(A_n)_{n:\mathbb{N}}$ is a sequence of sets in Ω and $A_n \downarrow A$ or $A_n \uparrow A$, then $A \in \mathcal{M}$. ($A_n \uparrow A$ means that $(A_n)_{n:\mathbb{N}}$ is a sequence of sets such that $A_{n+1} \supset A_n$ and $A = \bigcup_n A_n$.) It can be shown that a necessary and sufficient condition for a field $\mathcal{F}_0$ to he a σ-field is that it is a monotonic class.

A σ-field is *generated* by a collection of sets it it is the smallest σ-field which contains those sets. A set A is a *support* of a measure P if $P\,A = 1$.

Extension of measure

The following theorem is important for us because it implies that to prove equality of two measures defined on a σ-field $\mathcal{F}$ it suffices to prove that they agree on a subset of the sets in $\mathcal{F}$, namely on a field $\mathcal{F}_0$ which generates $\mathcal{F}$.

The Extension Theorem A probability measure on a field $\mathcal{F}_0$ has a unique extension to the σ-field $\mathcal{F}$ generated by $\mathcal{F}_0$. □

For a proof of this theorem see [Bi79]. It can also be shown that if the σ-field $\mathcal{F}$ is generated by a class $\mathcal{P}$ of subsets of Ω which is closed under finite intersectious, then to prove equality of two measures on $\mathcal{F}$ it suffices to show that they agree on $\mathcal{P}$.

Measurable functions

Given two measurable spaces $(\Omega, \mathcal{F})$ and $(\Omega', \mathcal{F}')$, a function $f : \Omega \to \Omega'$ is said to be *measurable* $\mathcal{F}/\mathcal{F}'$ if for all sets $A \in \mathcal{F}'$ the inverse image $f^{-1}A$ is an element of $\mathcal{F}$. If $(\Omega', \mathcal{F}') = (\mathbb{R}, \mathcal{R})$, i.e. the real line with the σ-field generated by the opeu intervals, then f is called a *random variable*. If the range of f is a finite set of points, f is called a *simple function* or *simple random variable* and can be uniquely written in the form

$$f = \sum_{i=1}^{n} a_i I_{A_i}$$

where $\{ a_i \mid 0 \leq i \leq n \}$ is the range of f, $A_i = f^{-1}a_i$, and I_{A_i} is the *indicator function*, defined as

$$I_A u \triangleq \begin{cases} 1 & \text{if } u \in A \\ 0 & \text{otherwise .} \end{cases}$$

We will need the following results about random variables, which we quote from [Sh84]:

Theorem 2.1.1 Any non-negative measurable function $f : \Omega \to \mathbb{R}^+$ is the limit of a monotone increasing sequence of non-negative simple functions. □

Theorem 2.1.2 If f and g are measurable functions: $\Omega \to \mathbb{R}$ and $k \in \mathbb{R}$ then each of the functions :

$$f + k,\ kf,\ f + g,\ fg$$

is measurable. □

Theorem 2.1.3 The limit $\lim_n f$ of a convergent sequence $(f_n)_{n:\mathbb{N}}$ of random variables is measurable (i.e. a random variable). □

To prove a function measurable it suffices to show that $f^{-1}A' \in \mathcal{F}$ for each $A' \in \mathcal{A}'$ where $\mathcal{A}'$ generates $\mathcal{F}'$. Also if f is measurable $\mathcal{F}/\mathcal{F}'$ and f' is measurable $\mathcal{F}'/\mathcal{F}''$ then the function $f\,;f'$ obtained by composing f and f' is measurable $\mathcal{F}/\mathcal{F}''$ [Bi79].

Transformation of measure

Given a measure P on $(\Omega, \mathcal{F})$ and an $\mathcal{F}/\mathcal{F}'$-measurable function f we can transform P into a measure P' on $(\Omega', \mathcal{F}')$ by setting

$$P'A \ \hat{=}\ P\ f^{-1}A$$

for any set $A \in \mathcal{F}'$. P' is a measure since $P'A$ is well defined for all sets $A \in \mathcal{F}'$ and conntable additivity of P' follows from that of P. P' is called the measure *induced* by f.

We use transformation functions to define most of the operators in our language. Often the laws that link different operators follow from the fact that different combinations of transformation functions are the same; obviously if $f\,;g = g\,;f$ then the measure induced by $f\,;g$ is the same as the measure induced by $g\,;f$.

Linear combination of measures

Lemma 2.1.4 If P and Q are probability measures and $0 \leq p \leq 1$ then the function $R : \mathcal{F} \to [0,1]$ defined by

$$R\ A \ \hat{=}\ p\ P\ A + (1-p)\ Q\ A$$

is also a measure. □

Proof Clearly $R\ \Omega = 1$, $R\ \emptyset = 0$ and $0 \leq R\ A \leq 1$ for all other sets $A \in \mathcal{F}$. It remains to show that R is countably additive: for a disjoint sequence $(A_n)_{n:\mathbb{N}}$ of sets

$$\begin{aligned} R(\bigcup A_n) &= p\ P(\bigcup A_n) + (1-p)\ Q(\bigcup A_n) \\ &= p \sum P\ A_n + (1-p) \sum Q\ A_n \\ &\qquad \text{by countable additivity of } P, Q \\ &= \sum (p\ P\ A_n + (1-p)\ Q\ A_n) \\ &= \sum R\ A_n. \end{aligned}$$

□

Integration

In connection with product measures it will prove useful to use the notation of integrals. Integration of a simple function f with respect to a measure P is defined by

$$\int f(u)P(du) \ \hat{=}\ \sum_{i=1}^{n} a_i\ P f^{-1} a_i.$$

The definition of the integral of an arbitrary non-negative measurable function $f : \Omega \to \mathbb{R}$ is based on the fact that every such function is the limit of a monotonic increasing sequeuce of simple functions.

$$\int f(u)P(du) \ \hat{=}\ \sup \{ \int s(u)P(du) \mid s \leq f, s \text{ a simple function} \}.$$

We will need the following simple form of a *change of variable*: if P' is a measure induced by the function $h : \Omega \to \Omega$ then

$$\int f(u)P'(du) = \int f(h(w))P(dw).$$

For instance the probability of a set of traces A can be written as the integral of I_A. Suppose that

$$\int I_A(u)P'(du) = \int I_A(h(w))P(dw).$$

Since $I_{h^{-1}A}(w) = I_A(h(w))$ this means that P' is an induced measure:

$$P'A = \int I_A(u)P'(du) = \int I_{h^{-1}A}(w)P(dw).$$

Weak Convergence

In chapter 4 we will construct the fixed point of a recursive equation as the limit of a convergent sequence of measures. The standard concept of convergence in the space of measures is weak convergence, which is usually defined as follows [Sh84]: Let Ω be a metric space under metric δ. Let $\mathcal{F}$ be the σ-field of subsets of Ω which is generated by the open sets with respect to δ. Let $(P_n)_{n:\mathbb{N}}$ be a sequence of probability measures on $(\Omega, \mathcal{F})$.

Definition 2.1.5 The sequence of probability measures $(P_n)_{n:\mathbb{N}}$ *converges weakly* to the probability measure P (notation $P_n \xrightarrow{w} P$) if

$$\int_\Omega f(x) P_n(dx) \;\rightarrow\; \int_\Omega f(x) P(dx)$$

for every function f in the class $\mathbb{C}(\Omega)$ of continuous bounded functions on Ω. □

Textbook examples like the law of large numbers motivate use of weak convergence, but we will wait until chapter 4 to give one arising from the semantics of probabilistic communicating processes. The following theorem which we quote from [Sh84] provides us with two alternative conditions which we will find more convenient to use than definition 2.1.5.

Theorem 2.1.6 The following statements are equivalent.

1. $P_n \xrightarrow{w} P$.
2. $\limsup P_n A \leq P A$ for every closed set A.
3. $\liminf P_n A \geq P A$ for every open set A.

□

Product Measure

Let $A \times B$ denote the Cartesian product of two sets:

$$A \times B \;\hat{=}\; \{(u, v) \mid u \in A \wedge v \in B\}.$$

Given two probability spaces $(\Omega_X, \mathcal{F}_X, P_X)$ and $(\Omega_Y, \mathcal{F}_Y, P_Y)$ we construct the product space $(\Omega_{XY}, \mathcal{F}_{XY}, P_{XY})$ as follows. The set Ω_{XY} consists of the pairs of points in $\Omega_X \times \Omega_Y$. The σ-field $\mathcal{F}_{XY}$ is generated by the *measurable rectangles* which are sets of the form $A \times B$ where $A \in \mathcal{F}_X$ and $B \in \mathcal{F}_Y$. These sets have probability

$$P_{XY} A \times B \;\hat{=}\; P_X A \; P_Y B.$$

This definition gives a countably additive function on the field of finite disjoint unions of measurable rectangles. By the extension theorem it extends uniquely to a measure on the σ-field $\mathcal{F}_{XY}$ which is generated by the measurable rectangles.

For general $E \in \mathcal{F}_{XY}$ this measure can be written as

$$P_{XY}E = \int P_X E_y P_Y(dy)$$

where we adopt the notation $E_y = \{x \mid (x, y) \in E\}$ so that E_y represents a section through the set E at y. If $E = A \times B$ then

$$E_y = \begin{cases} A & \text{if } y \in B \\ \emptyset & \text{otherwise} \end{cases}$$

and hence $P_X E_y = I_B(y) P_X A$ so that as required

$$P_{XY} A \times B = \int I_B(y) P_X A \; P_Y(dy) = P_X A \; P_Y B.$$

By Fubini's theorem the order of integration is reversible, i.e.

$$\int P_X E_y P_Y(dy) = \int P_Y E_x P_X(dx).$$

We will use $(P_X \times P_Y)$ as an alternative notation for the product measure. (This form is clearer if we distinguish measures not by snbscripts but by different upper case letters.)

Conditional probability measures

In chapter 7 we will model dependence on an environment by defining a process as a *conditional probability measure.*

Definition 2.1.7 A conditional probability measure (cpm) is a function of two parameters, $P : \mathcal{F} \times \Omega \to [0, 1]$, such that

- for fixed $y \in \Omega$ and varying $A \in \mathcal{F}$, $P(A, y)$ is a probability measure and
- for fixed $A \in \mathcal{F}$ and varying $y \in \Omega$, $P(A, y)$ is a $\mathcal{F}$-measurable random variable.

□

To give a semantics for a language in terms of conditional probability measures we also require products, transformations and linear combinations of conditional probability measures.

Product of cpm's

Let P_X and P_Y be two cpm's and let $f : \Omega \to \Omega$ and $g : \Omega \to \Omega$ be two measurable functions on traces. We define the product of P_X and P_Y with respect to f and g to be the function which, for given z, is the product measure of P_X given $f\,z$ and P_Y given $g\,z$.

Lemma 2.1.8 The function $P : \mathcal{F}_{XY} \times \Omega \to [0,1]$ defined for all $E \in \mathcal{F}_{XY}$ and $z \in \Omega$ by

$$P(E,z) \;\widehat{=}\; \int P_X(E_y, f\,z) P_Y(dy, g\,z)$$

is a conditional probability measure. □

Proof For fixed z, $f\,z$ and $g\,z$ are fixed, so that $P(E,z)$ is simply the product measure of P_X given $f\,z$ and P_Y given $g\,z$.

For fixed $A \in \mathcal{F}$. $P_X(A,z)$ and $P_Y(A,z)$ are random variables. Since f and g are measurable, it follows that $P_X(A, f\,z)$ and $P_Y(A, g\,z)$ are also random variables. Let $\mathcal{M}$ be the class of sets such that for fixed $E \in \mathcal{M}$ the function $P_{XY}(E,z)$ is a random variable. The class $\mathcal{M}$ contains the measurable rectangles:

$$P_{XY}(A \times B, z) = P_X(A, f\,z)\; P_Y(B, g\,z)$$

is a product of random variables and hence itself a random variable. Any set E in the field generated by the measurable rectangles can be expressed as a disjoint union, $\bigcup E_i$ say, of rectangles such that $P_{XY}(E,z) = \sum P_{XY}(E_i, z)$. As a sum of random variables is itself a random variable and $\mathcal{M}$ contains the field generated by the measurable rectangles. Thus it is enough to show that $\mathcal{M}$ is a monotonic class to deduce that $\mathcal{F}_{XY} \subseteq \mathcal{M}$. Suppose that $(E_n)_{n:\mathbb{N}}$ is a sequence of sets in $\mathcal{M}$ such that $E_n \downarrow E$. Then $P_{XY}(E_n, z) \downarrow P_{XY}(E, z)$ for all z . Thus $P_{XY}(E,z)$ is the limit of a sequence of random variables and hence itself a random variable. Therefore $E \in \mathcal{M}$. □

Transformation of cpm

Lemma 2.1.9 Let f, $g : \Omega \to \Omega$ be two $\mathcal{F}$-measurable functions. Given two cpm's P and Q set

$$P'(A,z) \;\widehat{=}\; I_{\mathrm{ran}f}(z)\; P(f^{-1}A, g\,z) + I_{(\mathrm{ran}f)^c}(z)\; Q(A,z).$$

for all $A \in \mathcal{F}$ and $z \in \Omega$. The function P' is a cpm. □

Proof Recall that $I_B(z)$ denotes the indicator function which has value 1 if $z \in B$ and 0 otherwise. For fixed B and variable z this is a random variable and for fixed z and variable B this is a point measure. So I is a cpm. Since for fixed z the function $P'(A, z)$ is either $P(f^{-1}A, g\,z)$ or $Q(A, z)$ and both P and Q are cpm's, P' is a probability measure for fixed z. For fixed A, P' is a sum of two random variables and hence itself a random variable. □

Sums of cpm's

Lemma 2.1.10 Let $\{S_i\}$ be a partition of Ω and $\{P_i\}$ a set of cpm's. Then the function defined $\forall A \in \mathcal{F}.\, y \in \Omega$ as

$$P(A, y) \;\widehat{=}\; \sum_i I_{S_i}(y)\; P_i(A, y)$$

is also a cpm. □

Proof For each y there exists exactly one i such that $y \in S_i$. This gives $P(A, y) = P_i(A, y)$ which is a cpm by definition of P_i. For fixed A, $P(A, y)$ is a sum of products of random variables which is again a random variable. □

We will give reference to some other standard results of probability, like the Borel-Cantelli lemmas and the law of large numbers, as and when they are needed.

2.2 Notation

In *CSP* [Hoa85] each process is parametrised by an *alphabet*, or set of actions which it can perform. We use a universal (non-empty, finite or countable) alphabet Σ instead, and where necessary, as in alphabetised parallel composition, explicitely restrict a process to a subset of Σ. We usually use the letters a or e for actions. and B, C or D for sets of actions.

Sequences of actions are called *traces*. The following is an informal summary of the notation we use for traces and operations on traces. (For the formal definitions see [Hoa85].) The notation is used both for finite aud infinite traces, unless otherwise stated.

$\langle\rangle$	the empty trace,
$\langle a \rangle$	the trace containing only a,
$t\,s$	concatenation of traces t and s (where t finite),
$\#t$	the length of a trace t ($\#t = \infty$ if t infinite),
$t_n, n < \#t$	$(n+1)^{\text{th}}$ element of a trace t (the first element is always t_0),
$t \lceil n$	restriction of a trace t to its first n actions,
$t \restriction B$	restriction of a trace t to actions in the set B,
$t \downarrow B$	the number of elements of B contained in t,
$t < u$	t is a proper prefix of u,
$t^n, n : \mathbb{N}$	a finite trace t repeated n times ($t^0 = \langle\rangle$),
t^ω	a finite trace t repeated infinitely many times,
$t/n, n \leq \#t$	t after n, i.e. t with its first n steps removed.

By B^* and B^ω we denote the set of finite and infinite traces respectively made up of elements of B. Usually we use the letters t, s for finite traces and the letters u, v for infinite traces.

From now on we use Ω to denote the set of infinite traces. We introduce a special 'unobservable' action τ to encode as infinite traces with a tail $\langle\tau\rangle^\omega$ all finite traces after which a process may terminate. So

$$\Omega \;\widehat{=}\; \Sigma^\omega \cup \{\, t\langle\tau\rangle^\omega \mid t \in \Sigma^* \,\}.$$

We use another special action, $\surd$, to mark the successful termination of a process. We write Σ_τ as shorthand for $\Sigma \cup \{\tau\}$ and

$$\Sigma_\tau^* \;\widehat{=}\; \Sigma^* \cup \{\, t\,s \mid t \in \Sigma^* \wedge s \in \{\tau\}^* \,\}.$$

We will need a restriction function $\rceil$ which adds a tail of τ's where $\lceil$ produces a finite trace:

$$\forall z \in \Omega \cdot z \rceil B \;\widehat{=}\; \begin{cases} z \lceil B & \text{if } z \downarrow B = \infty \\ (z \lceil B)\langle\tau\rangle^\omega & \text{otherwise .} \end{cases}$$

Given a trace $t \in \Sigma_\tau^*$, let

$$S(t) \;\widehat{=}\; \{\, u : \Omega \mid u > t \}$$

denote the set of infinite traces which are extensions of t. If t consists of a single element a we leave out the brackets and write $S\langle a \rangle$. Note that the only trace leading on from a τ is the tail of τ's: $S(t\langle\tau\rangle) = \{t\langle\tau\rangle^\omega\}$. Also if $t \in \Sigma^*$ then $S(t\tau)$ can be expressed as a difference of sets with τ-free prefixes:

$$S(t\langle\tau\rangle) \;=\; S(t) - \bigcup_{e \neq \tau} S(t\langle e \rangle).$$

Sets of traces with a common prefix belong to the family of *cylinder sets* which are sets defined by a predicate on a finite number of dimensions in an infinite-dimensional space; using a set of k distinct indices $\{n_i \mid 0 < i \leq k\}$ and a set

$H \subseteq \Sigma^k$, a cylinder set can be written as $\{ u : \Omega \mid (u_{n_1}, \ldots, u_{n_k}) \in H \}$, i.e. the traces in this set are constrained only on the k-tuples picked out by the index set.

From now on let $\mathcal{F}$ denote the σ-field generated by the sets of infinite traces with a common prefix. As a σ-field, $\mathcal{F}$ is closed under the formation of finite and countable unions. It also contains the empty set ($\emptyset = \Omega^c$) and is closed under finite and countable intersections because $A \cap B = (A^c \cup B^c)^c$. We use the symbol $\equiv$ to denote that two syntactic expressions are equivalent in that they have the same semantics.

Chapter 3

A Model Without External Choice

In this chapter we present the semantics for a small language which we call $PCSP_0$. The syntax of $PCSP_0$ contains a subset of the constructs of *CSP* [Hoa85]. There is no external choice, and parallel composition is restricted to fully synchronised or simple parallel composition, because alphabetised parallel composition would result in external choice between unsynchronised actions. Internal (or non-deterministic) choice has been turned into probabilistic choice by adding a subscript to indicate the probability with which the choice is made. Similarly interleaving now has a probability attached to it.

$$\begin{aligned} P \ ::= \ & STOP \mid SKIP \mid X \mid a \rightarrow P \mid P \ {}_p\sqcap \ Q \mid P \backslash B \mid f(P) \mid \\ & P \parallel Q \mid P \, ; Q \mid P \ {}_p\!|\!|\!| \ Q \mid \mu X \bullet P \mid \langle X_i = P_i \rangle. \end{aligned}$$

Clause X introduces variables from a set VAR; these are required for the treatment of mutual recursion presented in chapter 4. The semantics of each variable is determined by a *binding* which maps each variable to an element of the space PM of probability measures on $(\Omega, \mathcal{F})$. Let BND_0 be the domain of all bindings.

$$BND_0 \ \widehat{=} \ VAR \rightarrow PM.$$

The semantics of a $PCSP_0$ term P is a function of the set of free variables appearing in P. For example, the semantics of $a \rightarrow X$ is parametrised by $\rho[\![X]\!]$, the semantics of X in the current binding ρ. Given ρ we can associate $a \rightarrow X$ with a measure in PM. Thus the semantic function for terms must be of type

$$PCSP_0 \rightarrow BND_0 \rightarrow PM.$$

Rather than clogging our notation with an explicit symbol for this function we overload the meaning of the denotational brackets and simply write $[\![P]\!]\rho$ to denote

the semantics of a term P in a binding ρ. The semantics may be evaluated by associating each free variable X with its value $\rho[\![X]\!]$ in the current binding. We write $\rho[Y/X]$ to change the binding ρ by associating the variable X with a new measure Y:

$$\rho[Y/X][\![Z]\!] \;\hat{=}\; \begin{array}{ll} Y & \text{if } Z = X \\ \rho[\![Z]\!] & \text{otherwise .} \end{array}$$

This enables us to define syntactic substitution $P[Q/X]$, where all occurrences of the variable X are replaced by the term Q, as having semantics

$$[\![P[Q/X]]\!]\rho \;\hat{=}\; [\![P]\!]\rho[[\![Q]\!]\rho/X].$$

Even though we need free variables to be able to define mutual recursion, we will ultimately be interested only in terms which represent *processes*. These terms contain no free variables and are therefore independent of the current binding. Thus when defining a process we can omit the binding parameter. Also it turns out that up until recursion the parameter ρ is carried through the proof of every algebraic equivalence without ever changing. So for simplicity's sake we omit ρ in these proofs -- they could be made rigorous simply by inserting ρ to the right of each term in denotational brackets.

We now present the semantics of $PCSP_0$. Let A be an arbitrary set in $\mathcal{F}$.

3.1 Atoms

The process *STOP* deadlocks immediately, i.e. it never does anything. This behaviour corresponds to the point measure which gives probability 1 to the trace of unobservable actions and probability 0 to everything else:

$$[\![STOP]\!]\rho\ A \;\hat{=}\; \begin{cases} 1 & \text{if } \langle\tau\rangle^\omega \in A \\ 0 & \text{otherwise .} \end{cases}$$

As in *CSP* we distinguish between deadlock and successful termination, which is marked by the special action $\surd$. Once a process has performed this action it cannot do anything else (although potentially another process can take over). The process *SKIP* does nothing but terminate successfully.

$$[\![SKIP]\!]\rho\ A \;\hat{=}\; \begin{cases} 1 & \text{if } \langle\surd\rangle\langle\tau\rangle^\omega \in A \\ 0 & \text{otherwise .} \end{cases}$$

3.2 Prefixing

The expression $a \rightarrow P$ denotes a process which first performs the observable action a and then behaves as process P. We use the function $prefix_a$, which prefixes a trace with an a, to define the probability measure denoted by $a \rightarrow P$ as a transformation of the measure denoted by P.

$$[\![a \rightarrow P]\!]\rho\ A \ \hat{=}\ [\![P]\!]\rho\ prefix_a^{-1}A$$

where

$$prefix_a : \Omega \rightarrow \Omega$$
$$\forall\, u \in \Omega \ \cdot\ prefix_a(u) = \langle a \rangle u.$$

The following lemma shows that this is a valid definition.

Lemma 3.2.1 The function $prefix_a$ is measurable. □

Proof We show that the inverse image of each generating set is in $\mathcal{F}$. For a non-empty sequence $t \in \Sigma_\tau^*$ the set $S(t)$ has inverse image

$$prefix_a^{-1}S(t) \ = \ \begin{cases} S(t/1) & \text{if } t_0 = a \\ \emptyset & \text{otherwise .} \end{cases}$$

Also $prefix_a^{-1}\Omega = prefix_a^{-1}S\langle a \rangle \cup \bigcup_{e \neq a} prefix_a^{-1}S\langle e \rangle = \Omega$. □

Using the above expression for the inverse image of $S(t)$ we can write its probability as

$$[\![a \rightarrow P]\!]S(t) \ = \ \begin{cases} [\![P]\!]S(t/1) & \text{if } t_0 = a \\ 0 & \text{otherwise .} \end{cases}$$

So $a \rightarrow P$ must do a as its first step, and the probability of any further steps depends on P. This is analogous to the behaviour of $a \rightarrow P$ in other models of *CSP*.

3.3 Probabilistic Choice

We write $P\ {}_p\!\sqcap\ Q$ for a process which behaves like P with probability p and like Q with probability $1-p$. This corresponds to the weighted average of the measures of P and Q.

$$[\![P\ {}_p\!\sqcap\ Q]\!]\rho\ A \ \hat{=}\ p\ [\![P]\!]\rho\ A \ +\ (1-p)\ [\![Q]\!]\rho\ A.$$

It follows from lemma 2.1.4 that $[\![P \ {}_p\sqcap \ Q]\!]$ is a measure. Probabilistic choice satisfies similar laws as the choice operator in the traces model of *CSP*:

Lemma 3.3.1

Probabilistic choice is idempotent aud commutative.

L1 $P \ {}_p\sqcap \ P \equiv P.$

L2 $P \ {}_p\sqcap \ Q \equiv Q \ {}_{1-p}\sqcap \ P.$

A choice with probability 1 is certainty.

L3 $P \ {}_1\sqcap \ Q \equiv P.$

The associative law holds if the weights attached to the choices are adjusted appropriately. It is expressed most neatly in the following, slightly unusual, form:

L4 $(P \ {}_{p/(1-q)}\sqcap \ Q) \ {}_{1-q}\sqcap \ R \equiv (R \ {}_{q/(1-p)}\sqcap \ Q) \ {}_{1-p}\sqcap \ P.$

Prefixing distributes through choice.

L5 $a \rightarrow (P \ {}_p\sqcap \ Q) \equiv (a \rightarrow P) \ {}_p\sqcap \ (a \rightarrow Q).$

□

Proof Laws 1 to 3 follow immediately from the definitiou. The measures for both sides of law 4 expand to

$$p[\![P]\!]A + (1-p-q)[\![Q]\!]A + q[\![R]\!]A.$$

Prefixing distributes through probabilistic choice because

$$\begin{aligned}
[\![a \rightarrow (P \ {}_p\sqcap \ Q)]\!] \ A &= [\![P \ {}_p\sqcap \ Q]\!] \ prefix_a^{-1}A \\
&= p \ [\![P]\!] \ prefix_a^{-1}A + (1-p) \ [\![Q]\!] \ prefix_a^{-1}A \\
&= p \ [\![a \rightarrow P]\!] \ A + (1-p) \ [\![a \rightarrow Q]\!] \ A \\
&= [\![(a \rightarrow P) \ {}_p\sqcap \ (a \rightarrow Q)]\!] \ A.
\end{aligned}$$

□

Binary non-deterministic choice can be generalised to $\prod_{p_i} P_i$ where the P_i are $PCSP_0$ terms and the p_i are probabilities, that is $0 \leq p_i \leq 1$ and $\sum_i p_i = 1$.

3.4 Hiding

Hiding, or removing a set of observable actions from the traces, enables us to abstract from unnecessary detail in the behaviour of a process. The expression $P\backslash B$, where $B \subseteq \Sigma$, denotes a process which behaves like P without the actions in B. Using the function $hide_B$ which removes all actions in B from the traces, we define the semantics of $P\backslash B$ as a transformation of the measure denoted by P. The probability of a set of traces after hiding is the probability of all the traces containing actions in B which it could have stemmed from. So

$$[\![P\backslash B]\!]\rho\ A \ \hat{=}\ [\![P]\!]\rho\ hide_B^{-1} A$$

where

$$hide_B : \Omega \to \Omega$$
$$\forall u \in \Omega \cdot\ hide_B(u) = u \restriction B^c$$

Lemma 3.4.1 The function $hide$ is measurable. □

Proof Consider the inverse image of the generating set $S(t)$ where $t \in \Sigma_\tau^*$. If t contains any element of B then $hide_B^{-1} S(t) = \emptyset$. Otherwise suppose that t contains only visible actions. Then the inverse image of $S(t)$ consists of all the infinite traces which, after hiding of B, begin with t. Each such trace u must have a finite prefix s such that $hide_B(s) = t$, i.e. $hide_B(u) \geq t$ if and only if there exists an s such that $u > s$ and $hide_B(s) = t$.

$$\begin{aligned} hide_B^{-1} S(t) &= \{u \mid hide_B(u) > t\} \\ &= \bigcup_{hide_B(s)=t} S(s). \end{aligned}$$

Since there are only countably many finite traces the set $\bigcup S(s)$ is at worst a countable union of $\mathcal{F}$-sets and hence itself in $\mathcal{F}$.

If t ends in a tail of τ's we can write $S(t) = S(t'\langle\tau\rangle)$ where $t' = t \restriction \Sigma$. Since

$$S(t'\langle\tau\rangle) = S(t') - \bigcup_{e \neq \tau} S(t'\langle e\rangle)$$

it follows that

$$hide_B^{-1} S(t'\langle\tau\rangle) = hide_B^{-1} S(t') - \bigcup_{e \neq \tau} hide_B^{-1} S(t'\langle e\rangle)$$

which, as a difference of measurable sets, is also measurable. Thus the inverse image of every generating set is measurable. □

The argument concerning traces ending in a tail of τ's can be used for any transformation function. So for the remaining operators we only need to prove measurability for sets with τ-free prefixes.

Lemma 3.4.2

Hiding everything produces a process which does nothing.

L1 $P\backslash\Sigma \equiv STOP$.

Hiding nothing changes nothing.

L2 $P\backslash\emptyset \equiv P$.

Hiding does not affect a process which does nothing.

L3 $STOP\backslash B \equiv STOP$.

Hiding first one set of actions and then another is the same as hiding the union of both sets.

L4 $(P\backslash B)\backslash C \equiv P\backslash B \cup C$.

Hidden actions disappear; other actions are unaffected.

L5 $(a \rightarrow P)\backslash B \equiv \begin{cases} a \rightarrow P\backslash B & \text{if } a \notin B \\ P\backslash B & \text{otherwise .} \end{cases}$

Hiding distributes through probabilistic choice.

L6 $(P \,{}_p\!\sqcap\, Q)\backslash B \equiv P\backslash B \,{}_p\!\sqcap\, Q\backslash B$.

□

Proof For law 1 note that $hide_\Sigma u = \langle\tau\rangle^\omega$ for all u. Therefore

$$hide_\Sigma^{-1} A = \begin{cases} \Omega & \text{if } \langle\tau\rangle^\omega \in A \\ \emptyset & \text{otherwise .} \end{cases}$$

So

$$\begin{aligned} [\![P\backslash\Sigma]\!]\, A &= [\![P]\!] hide_\Sigma^{-1}\, A \\ &= \begin{cases} 1 & \text{if } \langle\tau\rangle^\omega \in A \\ 0 & \text{otherwise} \end{cases} \\ &= [\![STOP]\!]\, A. \end{aligned}$$

Law 2 follows from the fact that $hide_\emptyset = id$, the identity function on traces. For law 3 note that $\langle\tau\rangle^\omega \in A \Leftrightarrow \langle\tau\rangle^\omega \in hide_B^{-1} A$ so that

$$\begin{aligned}[\![STOP\backslash B]\!]\ A &= [\![STOP]\!]\ hide_B^{-1} A \\ &= \begin{cases} 1 & \text{if } \langle\tau\rangle^\omega \in A \\ 0 & \text{otherwise} \end{cases} \\ &= [\![STOP]\!]\ A.\end{aligned}$$

Law 4 holds because $hide_B\ ;\ hide_C = hide_{B\cup C}$ and law 5 because

$$prefix_a\ ;\ hide_B = \begin{cases} hide_B\ ;\ prefix_a & \text{if } a \notin B \\ hide_B & \text{otherwise .} \end{cases}$$

The proof of the law that hiding distributes through choice is similar to the proof that prefixing distributes through choice. □

3.5 Simple Parallel Composition

In simple parallel composition two processes must cooperate on every action that is performed. We would expect the probability that the parallel system $P\|Q$ performs an action to be the product of the probabilities with which the components P and Q perform this action. So it seems natural to define the measure for $P\|Q$ as a transformation of the product measure $([\![P]\!] \times [\![Q]\!])$. This transformation uses a function *par* which maps a pair of traces to the longest trace up to which they agree. If that is a finite trace it adds a tail of unobservable actions. This reflects the fact that for the parallel system to perform an infinite trace u both component processes must perform u. If the component processes set out to perform traces which differ after n steps the parallel system will deadlock at that point.

$$[\![P\|Q]\!]\rho\ A \ \hat{=}\ ([\![P]\!]\rho \times [\![Q]\!]\rho)\ par^{-1} A$$

where

$$par : \Omega \times \Omega \to \Omega$$

$$\forall\, u, v \in \Omega \ \cdot par(u,v) = \begin{cases} u & \text{if } u = v \\ (u\restriction n)\langle\tau\rangle^\omega & \text{if } u\restriction n = v\restriction n \wedge u_n \neq v_n. \end{cases}$$

Lemma 3.5.1 The function *par* is measurable. □

Proof Consider the inverse image of the set of extensions of a τ-free trace $t \in \Sigma^*$.

$$\begin{aligned} par^{-1} S(t) &= \{(u,v) \mid par(u,v) > t\} \\ &= \{(u,v) \mid (u = v \wedge u > t) \vee \\ &\qquad (\exists\, n \cdot u\restriction n = v\restriction n \wedge u_n \neq v_n \wedge (u\restriction n)\langle\tau\rangle^\omega > t)\} \\ &= S(t) \times S(t). \end{aligned}$$

By the same argument which we used in lemma 3.4.1 it follows that the set of extensions of a trace ending in τ also has a measurable inverse image. Thus *par* is measurable. □

Note that if t is τ-free then the inverse image of $S(t\langle\tau\rangle)$ can be written as

$$par^{-1}S(t\langle\tau\rangle) \;=\; S(t\langle\tau\rangle)\times S(t\langle\tau\rangle) \;\cup\; \bigcup_{e}\bigcup_{g\neq e} S(t\langle e\rangle)\times S(t\langle g\rangle)$$

where $e, g \in \Sigma_\tau$. So the probability of deadlock in a parallel system derives from the probability that the components deadlock individually or that they attempt to do different things.

As in *CSP*, parallel composition in $PCSP_0$ is not idempotent, as is shown by the two coin-tossing processes in parallel (example 5.3). The following lemma shows which laws do hold.

Lemma 3.5.2

Parallel composition is commutative and associative.

L1 $P\|Q \equiv Q\|P$.

L2 $P\|(Q\|R) \equiv (P\|Q)\|R$.

A process in parallel with *STOP* can do nothing.

L3 $P\|STOP \equiv STOP$.

Parallel composition distributes through probabilistic choice.

L4 $(P\ {}_p\!\sqcap\ Q)\|R \equiv P\|R\ {}_p\!\sqcap\ Q\|R$.

If two parallel processes are both prepared to perform the same action, they will synchronise in doing so.

L5 $(a \to P)\|(a \to Q) \equiv a \to (P\|Q)$.

If two parallel processes attempt to perform different actions, they deadlock.

L6 $a \neq b \Rightarrow (a \to P)\|(b \to Q) \equiv STOP$.

□

Proof Law 1 follows from the symmetry of par and Fubini's theorem. To prove that parallel composition is associative we show that $(id, par); par = (par, id); par$: For all $u, v, w \in \Omega$

$$
\begin{aligned}
&((id, par)\ ;\ par)(u, v, w)\\
&= \begin{cases} par(u, v) & \text{if } v = w \\ par(u, (v \restriction n)\langle\tau\rangle^\omega) & \text{if } v \restriction n = w \restriction n \wedge v_n \neq w_n \end{cases}\\
&= \begin{cases} u & \text{if } u = v = w \\ (u \restriction n)\langle\tau\rangle^\omega & \text{if } u \restriction n = v \restriction n = w \restriction n \wedge (u_n \neq v_n \vee v_n \neq w_n \vee u_n \neq w_n) \end{cases}\\
&= ((par, id)\ ;\ par)(u, v, w).
\end{aligned}
$$

To prove Law 3 we write

$$
\begin{aligned}
&[\![P \| STOP]\!] A\\
&= ([\![P]\!] \times [\![STOP]\!])\ par^{-1} A\\
&= ([\![P]\!] \times [\![STOP]\!])\ par^{-1} A \cap (\Omega \times \{\langle\tau\rangle^\omega\})\\
&\quad \text{since } ([\![P]\!] \times [\![STOP]\!])\ (\Omega \times \{\langle\tau\rangle^\omega\}) = 1\\
&= \begin{cases} 1 & \text{if } \langle\tau\rangle^\omega \in A \\ 0 & \text{otherwise} \end{cases}\\
&\quad \text{since } \langle\tau\rangle^\omega \in A \Leftrightarrow (\Omega \times \{\langle\tau\rangle^\omega\}) \subseteq par^{-1} A\\
&= [\![STOP]\!]\ A.
\end{aligned}
$$

Parallel composition distributes through probabilistic choice (law 4) because

$$
\begin{aligned}
&[\![P \| (Q\ {}_p\sqcap\ R)]\!]\ A\\
&= \int [\![P\ {}_p\sqcap\ Q]\!]\ (par^{-1} A)_z\ [\![R]\!]\ (dz)\\
&= p \int [\![P]\!]\ (par^{-1} A)_z\ [\![R]\!]\ (dz) + (1-p) \int [\![Q]\!]\ (par^{-1} A)_z\ [\![R]\!]\ (dz)\\
&= [\![P \| Q\ {}_p\sqcap\ P \| R]\!]\ A.
\end{aligned}
$$

Distributivity in the other direction is simply a consequence of symmetry. For law 5 it is straightforward to check that $(prefix_a, prefix_a); par = par; prefix_a$. To prove Law 6, note that $((prefix_a, prefix_c)\ ;\ par)(u, v) = par(\langle a\rangle u, \langle c\rangle v) = \langle\tau\rangle^\omega$ for all $u, v \in \Omega$. Thus the inverse of any set A through this transformation is $\Omega \times \Omega$ if A contains $\langle\tau\rangle^\omega$ and empty otherwise. Hence

$$
\begin{aligned}
&[\![(a \rightarrow P) \| (b \rightarrow Q)]\!]\ A\\
&= \begin{cases} [\![P]\!]\ \Omega\ [\![Q]\!]\ \Omega & \text{if } \langle\tau\rangle^\omega \in A \\ [\![P]\!]\ \emptyset\ [\![Q]\!]\ \emptyset & \text{otherwise} . \end{cases}\\
&= [\![STOP]\!]\ A.
\end{aligned}
$$

□

Simple parallel composition is very restrictive because it requires processes to synchronise on every action. Alphabetised parallel composition would allow some actions to be performed internally by one process without the participation of others. Unfortunately, this cannot be modelled as a transformation of measure because if two processes each set out to perform an internal action the two actions can happen in either order, and the pair of component traces beginning with these internal actions is related to more than one trace of the parallel system. So there is no function from pairs of component traces to system traces which could be used to induce a measure for the parallel system. We will investigate other ways of defining alphabetised parallel composition in chapters 6 and 7, but we cannot incorporate it into $PCSP_0$.

3.6 Sequential Composition

We denote sequeutial composition of two processes P and Q by $P;Q$. Like parallel composition, it is defined as a transformation of the product measure:

$$[\![P;Q]\!]\rho\ A \ \hat{=}\ ([\![P]\!]\rho \times [\![Q]\!]\rho)\ seq^{-1}A$$

where seq is a function which cuts the tail off its first argument at the $\surd$ and concatenates it with the second argument:

$$seq : \Omega\times\Omega \to \Omega$$
$$\forall u, v \in \Omega \cdot\ seq(u,v) = \begin{cases} u & \text{if } u \text{ } \surd\text{-free} \\ (u\lceil n)v & \text{if } u\lceil n \text{ } \surd\text{-free } \wedge\ u_n = \surd. \end{cases}$$

Lemma 3.6.1 The function seq is measurable. □

Proof For all $t \in \Sigma^*$

$$\begin{aligned} seq^{-1}S(t) &= \{(u,v) \mid (u > t \wedge u \text{ } \surd\text{-free}) \vee \\ &\qquad (\exists n \cdot u\lceil n \text{ } \surd\text{-free } \wedge\ u_n = \surd \wedge (u\lceil n)v > t)\} \\ &= \{(u,v) \mid (u\lceil \#t \text{ } \surd\text{-free } \wedge\ u > t) \vee \\ &\qquad (\exists 0 \le n < \#t \cdot u\lceil n \text{ } \surd\text{-free } \wedge\ u_n = \surd \wedge (u\lceil n)v > t)\}. \end{aligned}$$

So if t is $\surd$-free the inverse image $seq^{-1}S(t)$ contains all the pairs of sequences where the first sequence begins with a prefix of t followed by $\surd$ and the second sequence makes up the rest of t. It also contains the pairs of sequences where the first argument contains the whole of t (not necessarily followed by $\surd$), and the second sequence is arbitrary:

$$seq^{-1}S(t) = \bigcup_{n=0}^{\#t-1} S((t\lceil n)\langle\surd\rangle)\times S(t/n) \cup S(t)\times\Omega \qquad \text{if } t \text{ } \surd\text{-free}.$$

If t does contain a $\surd$ then it must stem from the second argument of *seq* because *seq* removes the first $\surd$. So

$$seq^{-1}S(t) \;=\; \bigcup_{n=0}^{k} S((t\restriction n)\langle\surd\rangle) \times S(t/n) \qquad \text{if } t\restriction k \text{ } \surd\text{-free} \wedge t_k = \surd.$$

In either case the inverse image is a finite union of measurable rectangles and thus a measurable set. Hence *seq* is measurable. □

Lemma 3.6.2

SKIP is the identity of sequential composition and *STOP* is the zero.

L1 $SKIP\,;P \;\equiv\; P\,;SKIP \;\equiv\; P.$

L2 $STOP\,;P \;\equiv\; STOP.$

Sequential composition is associative.

L3 $(P\,;Q)\,;R \;\equiv\; P\,;(Q\,;R).$

It distributes through probabilistic choice in both directions.

L4 $(P \;{}_p\!\sqcap\; Q)\,;R \;\cong\; (P\,;R) \;{}_p\!\sqcap\; (Q\,;R).$

L5 $P\,;(Q \;{}_p\!\sqcap\; R) \;\equiv\; (P\,;Q) \;{}_p\!\sqcap\; (P\,;R).$

Prefixing and sequential composition can be performed in either order.

L6 $(a \to P)\,;Q \;\equiv\; a \to (P\,;Q).$

Sequential composition distributes through hiding.

L7 If $\surd \notin B$ then $(P\,;Q)\backslash B \;\equiv\; (P\backslash B)\,;(Q\backslash B).$

□

Proof To prove Law 1 we first deal with $SKIP\,;P$.

$$\begin{aligned}
&(\llbracket SKIP\rrbracket \times \llbracket P\rrbracket)\; seq^{-1}A\\
&\quad= (\llbracket SKIP\rrbracket \times \llbracket P\rrbracket)\; (seq^{-1}A \cap \{\langle\surd\rangle\langle\tau\rangle^{\omega}\} \times \Omega)\\
&\qquad \text{since } (\llbracket SKIP\rrbracket \times \llbracket P\rrbracket)\; \{\langle\surd\rangle\langle\tau\rangle^{\omega}\} \times \Omega = 1\\
&\quad= (\llbracket SKIP\rrbracket \times \llbracket P\rrbracket)\; \{\langle\surd\rangle\langle\tau\rangle^{\omega}\} \times A\\
&\qquad \text{since } \forall u \in \Omega \cdot seq(\langle\surd\rangle\langle\tau\rangle^{\omega}, u) = u\\
&\quad= \llbracket SKIP\rrbracket\; \{\langle\surd\rangle\langle\tau\rangle^{\omega}\}\; \llbracket P\rrbracket\; A\\
&\quad= \llbracket P\rrbracket\; A.
\end{aligned}$$

Now consider $P\,;SKIP$. If $t \in \Sigma_\tau^*$ is $\surd$-free we have

$$\begin{aligned}[\![P\,;SKIP]\!]\,S(t) &= \sum_{n=0}^{\#t-1} [\![P]\!]\,S((t\restriction n)\langle\surd\rangle)\,[\![SKIP]\!]\,S(t/n) + [\![P]\!]\,S(t)\\ &= [\![P]\!]\,S(t)\end{aligned}$$

since if $0 \le n < \#t$ then $[\![SKIP]\!]\,S(t/n) = 0$. If t_k is the first $\surd$ in t then

$$\begin{aligned}[\![P\,;SKIP]\!]\,S(t) &= \sum_{n=0}^{k} [\![P]\!]\,S((t\restriction n)\langle\surd\rangle)\,[\![SKIP]\!]\,S(t/n)\\ &= [\![P]\!]\,S((t\restriction k)\langle\surd\rangle)\,[\![SKIP]\!]\,S(t/k).\end{aligned}$$

We will prove in lemma 3.6.3 that no process can ever do anything visible after terminating. It follows that if $t_k = \surd$ then

$$[\![P]\!]\,S(t) = \begin{cases} [\![P]\!]\,S((t\restriction k)\langle\surd\rangle) & \text{if } t/(k+1) < \langle\tau\rangle^\omega \\ 0 & \text{otherwise .}\end{cases}$$

Also $[\![SKIP]\!]\,S(t/k) = 1$ if $t/(k+1) < \langle\tau\rangle^\omega$ and 0 otherwise. So for a trace t which contains $\surd$ as its $(k+1)^{\text{th}}$ element

$$[\![P]\!]\,S((t\restriction k)\langle\surd\rangle)\,[\![SKIP]\!]\,S(t/k) = [\![P]\!]\,S(t).$$

So the second half of law 1 holds, too.

To prove Law 2 we use the fact that $\{\langle\tau\rangle^\omega\} \times \Omega$ is the support of the product measure and hence

$$\begin{aligned}[\![STOP\,;P]\!]\,A &= ([\![STOP]\!] \times [\![P]\!])\,seq^{-1}A\\ &= ([\![STOP]\!] \times [\![P]\!])\,(seq^{-1}A \cap \{\langle\tau\rangle^\omega\} \times \Omega).\end{aligned}$$

Since $seq^{-1}\langle\tau\rangle^\omega = \{\langle\surd\rangle\langle\tau\rangle^\omega\} \times \{\langle\tau\rangle^\omega\} \cup \{\langle\tau\rangle^\omega\} \times \Omega$ the intersection is non-empty if and only if $\langle\tau\rangle^\omega \in A$. So

$$\begin{aligned}[\![STOP\,;P]\!]\,A &= \begin{cases} 1 & \text{if } \langle\tau\rangle^\omega \in A \\ 0 & \text{otherwise}\end{cases}\\ &= [\![STOP]\!]\,A.\end{aligned}$$

Associativity of sequential composition follows if we can show that $(seq, id)\,;seq = (id, seq)\,;seq$. Given $u, v, w \in \Omega$ we have

$$seq(seq(u,v),w) = \begin{cases} seq(u,w) & \text{if } u \text{ } \surd\text{-free} \\ seq((u\restriction n)v, w) & \text{if } u\restriction n \text{ } \surd\text{-free } \wedge\ u_n = \surd\end{cases}$$

$$
\begin{aligned}
&= \begin{cases} u & \text{if } u \text{ } \surd\text{-free} \\ (u \lceil n)v & \text{if } u \lceil n \text{ } \surd\text{-free } \wedge u_n = \surd \wedge v \text{ } \surd\text{-free} \\ (u \lceil n)(v \lceil m)w & \text{if } u \lceil n \text{ } \surd\text{-free } \wedge u_n = \surd \wedge v \lceil m \text{ } \surd\text{-free } \wedge v_m = \surd \end{cases} \\
&= \begin{cases} u & \text{if } u \text{ } \surd\text{-free} \\ (u \lceil n)seq(v, w) & \text{if } u \lceil n \text{ } \surd\text{-free } \wedge u_n = \surd \end{cases} \\
&= seq(u, seq(v, w)).
\end{aligned}
$$

The proofs of Laws 4 and 5 follow along the same lines as the corresponding distributivity laws for parallel composition. For Law 6 it is easily checked that $(prefix_a, id)\,;\, seq = seq\,;\, prefix_a$. To prove Law 7 we show that $seq\,;\, hide_B = (hide_B, hide_B)\,;\, seq$. For all $u, v \in \Omega$

$$
\begin{aligned}
&(seq\,;\, hide_B)(u, v) \\
&= \begin{cases} hide_B u & \text{if } u \text{ } \surd\text{-free} \\ hide_B((u \lceil n)v) & \text{if } (u \lceil n) \text{ } \surd\text{-free } \wedge u_n = \surd. \end{cases} \\
&= \begin{cases} hide_B u & \text{if } u \text{ } \surd\text{-free} \\ ((hide_B u) \lceil m)(hide_B v) & \text{if } (hide_B u) \lceil m \text{ } \surd\text{-free } \wedge (hide_B u)_m = \surd. \end{cases} \\
&= ((hide_B, hide_B)\,;\, seq)(u, v).
\end{aligned}
$$

□

A process which terminates can never do anything else. Accordingly, the probability of a visible action happening after $\surd$ ought to be zero. This is expressed by the *termination constraint*: let U be the set of traces which contain a visible event after $\surd$.

$$U \;\hat{=}\; \{t\langle\surd\rangle u \mid t \text{ } \surd\text{-free } \wedge\, u \neq \langle\tau\rangle^\omega\}.$$

The following lemma asserts that all the measures representing $PCSP_0$-processes assign this set probability zero.

Lemma 3.6.3 $\forall P \in PCSP_0 \cdot [\![P]\!]\, U = 0.$ □

Proof We use structural induction. Clearly

$$[\![STOP]\!]\, U \;=\; [\![SKIP]\!]\, U \;=\; 0.$$

Consider now the operators which are transformations of measure and suppose that their arguments satisfy the constraint. A violation of the constraint can arise only if the inverse image of U through the transformation function contains traces outside U, because only they can have non-zero probability. However, it is easily checked that if f is any funtion defined so far (id, $prefix_a$, $hide_B$, par or seq), or one of the functions defined in the next two sections ($interleave$ or a relabelling function) then

$$f\ U^e \subseteq U^e$$

or, if f is binary,

$$f\ U^e \times U^e \subseteq U^e.$$

So all the transformation functions defined in this model preserve the termination constraint. Probabilistic choice preserves the constraint because the sum of two null-sets is again a null-set. It will be shown in chapter 4 that a recursively defined process is the limit of a sequence of iterates which consist of some combination of the above functions applied to $STOP$ a finite number of times. So the termination constraint is satified by each iterate and preserved in the limit. □

3.7 Prioritised Interleaving

Interleaving is similar to completely unsynchronised parallel composition in that the ordering of actions by different processes is entirely arbitrary. We will give a definition for the interleaving operator which works because it makes an assumption about this ordering, namely that we know the asymptotic frequency of actions by P and Q in the interleaved traces. This can be interpreted as knowledge about the relative speed of the component processes or, equivalently, that each process is scheduled some fixed proportion of the time.

We write $P\ {}_p|||\ Q$ to indicate that P and Q are interleaved in such a way that at each step P has a chance p of performing the next action. More succinctly, we say that P and Q are interleaved with P having priority p. If P deadlocks, Q proceeds on its own (irrespective of P's priority) and vice versa. If both P and Q deadlock the whole system deadlocks. If P has priority 1 then Q can only do something if P deadlocks. This is similar to the notion of process priority which [SS90] model in the context of $PCCS$. The system terminates successfully only when both P and Q are prepared to terminate. If only one process is prepared to terminate, then the other process takes over until it, too, can terminate. In effect, the action $\surd$ is the only action on which the processes must synchronise.

The semantics of this operator involves a transformation function *interleave* and a coin-tossing process $T(p)$.

$$[\![P\ {}_p|||\ Q]\!]\rho\ A \ \widehat{=}\ ([\![P]\!]\rho \times [\![Q]\!]\rho \times T(p))\ interleave^{-1} A.$$

The process $T(p)$ chooses between a 0 and a 1 with probability p and $1-p$ respectively at each step. Once we have defined recursion we will be able to write

$$T(p) \ \widehat{=}\ [\![\mu X \bullet ((0 \rightarrow X)\ {}_p\sqcap\ (1 \rightarrow X))]\!].$$

For our present purpose it suffices to know that for any τ-free trace $t \in \Sigma_\tau^*$ which contains k 0's we have

$$T(p)\ S(t) \quad \hat{=} \quad p^k(1-p)^{\#t-k}.$$

As long as neither P nor Q have deadlocked or terminated, P is allowed to make a step whenever $T(p)$ chooses 0 and Q is allowed a step whenever $T(p)$ chooses 1. The function *interleave* takes two traces of actions and interleaves them as determined by a trace of 0's and 1's, i.e. it is of type

$$\Omega \times \Omega \times \mathbb{B}^\omega \to \Omega.$$

Let $d \in \mathbb{B}^\omega$ and suppose that u and v are τ- and $\surd$-free up to at least $d\downarrow 0$ and $d\downarrow 0$ respectively. Then there exists a unique sequence of pairs of booleans and actions, such that the sequence of booleans is d and the sequence of actions labelled 0 is u and the sequence of actions labelled 1 is v. To express this formally let *zip* be the function which transforms a pair of sequences into a sequence of pairs, and let π_1, π_2 be projection functions such that for any two sequences l, r we have $\pi_1(zip(l,r)) = l$ and $\pi_2(zip(l,r)) = r$. Then for u and v as above we define

$$\begin{aligned} interleave(u,v,d) \quad &= \quad \pi_2\, z \\ &\qquad \text{where } z \in (\mathbb{B}\times\Sigma)^\omega \\ &\qquad\qquad \wedge\ \pi_1\, z = d \\ &\qquad\qquad \wedge\ \pi_2(z\restriction\{0\}\times\Sigma) \leq u \\ &\qquad\qquad \wedge\ \pi_2(z\restriction\{1\}\times\Sigma) \leq v \end{aligned}$$

If u is τ- and $\surd$-free only up to some up $n < d\downarrow 0$ and $u_n = \surd$ and v is τ- and $\surd$-free up to that point then the interleaved trace follows d until jnst before u_n is chosen and continues as the remainder of v.

$$\begin{aligned} interleave(u,v,d) \quad &= \quad (\pi_2\, z)(v/((\pi_1\, z)\downarrow 1) \\ &\qquad \text{where } z \in (\mathbb{B}\times\Sigma)^* \\ &\qquad\qquad \wedge\ (\pi_1\, z)\langle 0\rangle < d \\ &\qquad\qquad \wedge\ \pi_2(z\restriction\{0\}\times\Sigma)\langle\surd\rangle < u \\ &\qquad\qquad \wedge\ \pi_2(z\restriction\{1\}\times\Sigma) < v \end{aligned}$$

If everything is as in the last case except that $u_n = \tau$ then the interleave trace cannot terminate successfully. We therefore define

$$\begin{aligned} interleave(u,v,d) \quad &= \quad (\pi_2\, z)(v/((\pi_1\, z)\downarrow 1)\restriction\{\surd\}^c) \\ &\qquad \text{where } z \in (\mathbb{B}\times\Sigma)^* \\ &\qquad\qquad \wedge\ (\pi_1\, z)\langle 0\rangle < d \\ &\qquad\qquad \wedge\ \pi_2(z\restriction\{0\}\times\Sigma)\langle\tau\rangle < u \\ &\qquad\qquad \wedge\ \pi_2(z\restriction\{1\}\times\Sigma) < v \end{aligned}$$

The cases where the first τ or $\surd$ to be chosen by d stems from v are treated accordingly.

Lemma 3.7.1 The function *interleave* is measurable. □

Proof Let $t \in \Sigma^*$ be τ-free. Then

$$
\begin{aligned}
&interleave^{-1}S(t)\\
&= \bigcup_{d\in\mathbb{B}^{\#t}} S(\pi_2(zip(d,t)\restriction(\{0\}\times\Sigma)))\\
&\qquad \times S(\pi_2(zip(d,t)\restriction(\{1\}\times\Sigma)))\\
&\qquad \times S(d)\\
&\quad \cup \bigcup_{n=0}^{\#t-1} \bigcup_{d\in\mathbb{B}^n}\\
&\qquad S(\pi_2(zip(d,t)\restriction(\{0\}\times\Sigma))\langle\tau\rangle) \cup S(\pi_2(zip(d,t)\restriction(\{0\}\times\Sigma))\langle\surd\rangle)\\
&\qquad\quad \times S(\pi_2(zip(d,t)\restriction(\{1\}\times\Sigma))(t/n))\\
&\qquad\quad \times S(d\langle 0\rangle)\\
&\quad \cup\ S(\pi_2(zip(d,t)\restriction(\{0\}\cup\Sigma))(t/n))\\
&\qquad \times S(\pi_2(zip(d,t)\restriction(\{1\}\times\Sigma))\langle\tau\rangle) \cup S(\pi_2(zip(d,t)\restriction(\{1\}\times\Sigma))\langle\surd\rangle)\\
&\qquad \times S(d\langle 1\rangle)
\end{aligned}
$$

As a countable union of sets of traces with a common prefix the inverse image of $S(t)$ is a measurable set. If t ends in $\surd$ the only difference is that the component traces ending in τ are not in the inverse image. A set of traces with a prefix that is not τ-free can be expressed as the difference of sets with τ-free prefixes and is therefore also measurable. Heuce the function *interleave* is measurable. □

Using the above expression for $interleave^{-1}S(t)$ we can give an explicit expression for its probability. Each term in the nnion over all $d \in \mathbb{B}^{\#t}$ has probability

$$p^{\#\mathbb{0}}(1-p)^{\#\mathbb{1}}[\![P]\!]S(\pi_2(zip(d,t)\restriction(\{0\}\times\Sigma)))\ [\![Q]\!]S(\pi_2(zip(d,t)\restriction(\{1\}\times\Sigma))) \quad (3.1)$$

Similarly for the other terms. Note that $p = 1$ reduces the sum over all these terms to

$$[\![P]\!]\,S(t) + \sum_{n=0}^{\#t-1} [\![P]\!]\,S((t\restriction n)\langle\surd\rangle) \cup S((t\restriction n)\langle\tau\rangle)\ [\![Q]\!]\,S(t/n).$$

So, as mentioned earlier, if P has priority 1 then Q can only do something if P deadlocks or termiuates. Interleaving satisfies the following laws:

Lemma 3.7.2

Interleaving is commutative, associative and distributes through choice.

L1 $P \,{}_p|||\, Q \equiv Q \,{}_{1-p}|||\, P$.

L2 $(P \,{}_p\sqcap\, Q) \,{}_q|||\, R \equiv (P \,{}_q|||\, R) \,{}_p\sqcap\, (Q \,{}_q|||\, R)$.

L3 $(P \,{}_{p/1-q}|||\, Q) \,{}_{1-q}|||\, R \equiv (R \,{}_{q/1-p}|||\, Q) \,{}_{1-p}|||\, P$.

The process *SKIP* leaves the remaining component to run on its own.

L4 $SKIP \,{}_p|||\, P \equiv P$ (even if $p = 1$!).

L5 $(a \to P) \,{}_p|||\, (b \to Q) \equiv (a \to (P \,{}_p|||\, (b \to Q)) \,{}_p\sqcap\, (b \to ((a \to P) \,{}_p|||\, Q)))$.

□

Proof Commutativity is obvious, and distributivity through choice can be proved in the same way as the corresponding law for parallel composition.

Associativity can be proved by induction. The base cases are the probabilities of $S\langle\rangle$, $S\langle\tau\rangle$ and $S\langle\surd\rangle$. Assuming that the law holds for any $S(t)$ where $t \in \Sigma_\tau^*$ it can be shown to hold for $S(\langle a\rangle t)$ by expanding the inverse image of $S(\langle a\rangle t)$ twice using formula 3.1 and regrouping the resulting terms.

For law 4 note that $\forall\, u \in \Omega, d \in \mathbb{B}^\omega \cdot interleave(\langle\surd\rangle\langle\tau\rangle^\omega, u, d) = u$. Also the product measure $([\![SKIP]\!] \times [\![P]\!] \times T(p))$ has support $\{\langle\surd\rangle\langle\tau\rangle^\omega\} \times \Omega \times \Omega$. Therefore

$$
\begin{aligned}
&[\![SKIP_p \,|||\, P]\!]\, A \\
&= ([\![SKIP]\!] \times [\![P]\!] \times T(p))\,(interleave^{-1} A \cap \{\langle\surd\rangle\langle\tau\rangle^\omega\} \times \Omega \times \Omega) \\
&= ([\![SKIP]\!] \times [\![P]\!] \times T(p))\,\{\langle\surd\rangle\langle\tau\rangle^\omega\} \times A \times \Omega \\
&= [\![P]\!]\, A.
\end{aligned}
$$

For law 5 it is easy to check the following two identities:

$$
\begin{aligned}
(prefix_a, id, prefix_0)\,;\, interleave &= interleave\,;\, prefix_a \\
(id, prefix_b, prefix_1)\,;\, interleave &= interleave\,;\, prefix_b.
\end{aligned}
$$

Also, it follows from the recursive definition of $T(p)$ and law 1 of the recursion laws (4.2.9) that

$$
T(p)\; A = p\; T(p)\; prefix_0^{-1} A + (1-p)\; T(p)\; prefix_1^{-1} A.
$$

Using these facts we can write

$$
\begin{aligned}
&[\![(a \to P) \,{}_p|||\, (b \to Q)]\!] A \\
&= p\,([\![P]\!] \times [\![b \to Q]\!] \times T(p))\,(prefix_a^{-1}, id, prefix_0^{-1})(interleave^{-1} A) \\
&\quad + (1-p)\,([\![a \to P]\!] \times [\![Q]\!] \times T(p))\,(id, prefix_b^{-1}, prefix_1^{-1})(interleave^{-1} A) \\
&= p\,([\![P]\!] \times [\![b \to Q]\!] \times T(p))\; interleave^{-1}(prefix_a^{-1} A) \\
&\quad + (1-p)\,([\![a \to P]\!] \times [\![Q]\!] \times T(p))\; interleave^{-1}(prefix_b^{-1} A) \\
&= p\,[\![P \,{}_p|||\, (b \to Q)]\!]\; prefix_a^{-1} A \\
&\quad + (1-p)\,[\![(a \to P) \,{}_p|||\, Q]\!]\; prefix_b^{-1} A \\
&= [\![(a \to (P \,{}_p|||\, (b \to Q)) \,{}_p\sqcap (b \to ((a \to P) \,{}_p|||\, Q)))]\!]\; A.
\end{aligned}
$$

□

3.8 Relabelling

Let $f : \Sigma_\tau \to \Sigma_\tau$ be a function which relabels visible events but does not affect τ or $\surd$:

$$
\begin{aligned}
a = \tau \quad &\Leftrightarrow \quad f(a) = \tau \\
a = \surd \quad &\Leftrightarrow \quad f(a) = \surd.
\end{aligned}
$$

Lift f to sequences:

$$\forall u \in \Omega\ \forall i \in \mathbb{N} \cdot f(u)_i = f(u_i).$$

Then f can be used to define a probability measure

$$[\![f(P)]\!]\rho\, A \;\widehat{=}\; [\![P]\!]\rho\, f^{-1} A.$$

Lemma 3.8.1 The function f is measurable. □

Proof The function f applied to a trace does not affect the length of the trace. Thus the inverse image $f^{-1}S(t)$ is of the form $\bigcup_s S(s)$ where $\#s = \#t$ and $f(s) = t$. This is a measurable set. □

Relabelling satisfies the following laws:

Lemma 3.8.2

A process which does nothing remains unchanged by relabelling.

L1 $f(STOP) \equiv STOP$.

Relabelling a process first by one function and then another is the same as relabelling a process with the combined relabelling function.

L2 $f(g(P)) \equiv (g;f)P$.

Relabelling distributes through the following operators:

L3 $f(a \rightarrow P) \equiv f(a) \rightarrow f(P)$.

L4 $f(P \,{}_p\!\sqcap\, Q) \equiv f(P) \,{}_p\!\sqcap\, f(Q)$.

L5 $f(P \parallel Q) \equiv f(P) \parallel f(Q)$ if f is 1-1.

L6 $f(P;Q) \equiv f(P);f(Q)$.

L7 $f(P \,{}_p\!|||\, Q) \equiv f(P) \,{}_p\!|||\, f(Q)$.

□

Proof Law 1 holds because

$$[\![f(STOP)]\!]\, A = [\![STOP]\!]\, (f^{-1}A \cap \{\langle\tau\rangle^\omega\}) = [\![STOP]\!]\, A.$$

Law 2 is obvious. Law 3 holds because $prefix_a ; f = f ; prefix_{f(a)}$. Law 4 follows because

$$[\![f(P \,{}_p\!\sqcap\, Q)]\!]\, A = p\,[\![P]\!]f^{-1}A + (1-p)\,[\![Q]\!]f^{-1}A = [\![f(P) \,{}_p\!\sqcap\, f(Q)]\!]\, A.$$

Laws 5 to 7 hold because

$$\begin{array}{rcll} par\,;f & = & (f,f)\,;par & \text{if } f \text{ is 1-1} \\ seq\,;f & = & (f,f)\,;seq & \text{since } (a=\checkmark \;\Leftrightarrow\; f(a)=\checkmark) \\ interleave\,;f & = & (f,f,id)\,;interleave. & \end{array}$$

□

Chapter 4

Recursion

In this chapter we introduce operators for single aud mutual recursion in $PCSP_0$. The semantics of a recursive definition relies on the fact that the sequence of increasingly many unfoldiugs of the recursion converges. In the first section of this chapter we define weak convergence in the space PM of probability measures on $(\Omega, \mathcal{F})$ and show that a stronger concept of convergence wonld be unsuitable. In other models of CSP as well as other languages convergence is defined either with respect to a partial order (as in [Hoa85] and [JP89]) or with respect to a metric (as in [ReR88] and [DS91]). In both cases a fixed point theorem exists which yields a sufficient condition for the validity of a recursive definition which is easy to check.

A partial order on probability measures can easily be defined if the underlying space is ordered. For instance, given two measures P, Q on $(\mathbb{R}, \mathcal{R})$ we could define P to be below Q if $\forall x : \mathbb{R} \cdot P\{-\infty, x\} \leq Q(-\infty, x]$. However, the space of infinite sequences of actions is not ordered in a way which would have an intuitive appeal. [JP89] solve the problem by basing the semantics of their probabilistic language on *evaluations* rather than measures. Evaluations are like measures, but are defined only on a restricted class of sets and need not have total mass 1. One evaluation is defined to be below another if the "probabilities" assigned by the former are always less than those assigned by the latter. However, as the authors remark, it is more natural to use measures than evaluations. This is what we will do.

In the second section of this chapter we will define a metric and show that convergence respect to this metric is the same as weak convergence. This will enable us to take (almost) the standard approach towards establishing a sufficient condition for the validity of single recursion. In the third section we extend this approach to mutual recursion. In the last section we establish proof rules for recursion induction.

4.1 Weak Convergence

To define weak convergence in the space PM in the sense of definition 2.1.5 we use a metric δ on sequences which depends on the length of the longest prefix up to which they agree:

$$\forall u, v \in \Omega \cdot \delta(u,v) = \min\{2^{-n} \mid u\lceil n = v\lceil n\}.$$

The open balls in this space are the sets with fixed prefixes. Taking finite and countable unions as well as finite intersections of these sets yields the cylinder sets as open sets which, as required, are the generating sets of $\mathcal{F}$. They are actually clopen, because the complement of a cylinder set is also a cylinderset. Therefore it follows from theorem 2.1.6 that a sequence $(P_n)_{n:\mathbb{N}}$ of measures in PM converges weakly to a measure P only if for all cylinder sets A

$$\limsup P_n A = \liminf P_n A = \lim P_n A = P A.$$

To see why a concept stronger than weak convergence would be unsuitable consider $\mu X \bullet a \rightarrow X$. From our understanding of standard CSP we expect this to denote the process which performs infinitely many a's. In the probabilistic model this is the point measure

$$P A = \begin{cases} 1 & \text{if } \langle a\rangle^\omega \in A \\ 0 & \text{otherwise .} \end{cases}$$

We also expect $\mu X \bullet a \rightarrow X$ to be the limit of the sequence (P_n) where the process P_n performs n a's and then stops:

$$P_n A = \begin{cases} 1 & \text{if } \langle a\rangle^n\langle\tau\rangle^\omega \in A \\ 0 & \text{otherwise .} \end{cases}$$

The sequence $\{P_n\}$ converges on all sets with fixed prefixes:

$$\lim_n P_n S(t) = \begin{cases} 1 & \text{if } t < \langle a\rangle^\omega \\ 0 & \text{otherwise .} \end{cases}$$

However, it does not converge on all $A \in \mathcal{F}$. Consider the probabilities assigned by the P_n to the singleton set which contains just the infinite seqnence of a's. Writing $\{\langle a\rangle^\omega\} = \bigcap_k S\langle a\rangle^k$ and using the fact that $\forall k > n \cdot P_n S\langle a\rangle^k = 0$ we have

$$\lim_n P_n \left(\bigcap_k S\langle a\rangle^k\right) = 0.$$

Not only is this different from $P\{\langle a\rangle^\omega\}$, but it also means that the pointwise limit of the sequence of P_n's assigns 0 to all sets in $\mathcal{F}$ and thus fails to be a measure at all. By contrast, if we use weak convergence then because $\{\langle a\rangle^\omega\}$ is a closed set all that is required according to theorem 2.1.6 is that

$$\limsup P_n \{\langle a\rangle^\omega\} \leq P \{\langle a\rangle^\omega\}$$

which is true.

4.2 Single Recursion

Let P be a term possibly containing the free variable X. We write $\mu X \bullet P$ to denote a process that behaves as P with X representing a recursive invocation of the process.

To give a semantics to this expression first consider the semantics of P with a binding ρ. If we regard $\rho[\![X]\!]$ as being variable, $[\![P]\!]\rho$ becomes a function whose argument is the measure to be bound to X:

Definition 4.2.1 If P is a $PCSP_0$ term possibly containing the free variable X then

$$M(X,P)\rho \;\hat{=}\; \lambda Y \cdot [\![P]\!]\rho[Y/X].$$

□

Any free variables other than X are bound by ρ as usual. We can now give the semantics of the recursion operator:

$$[\![\mu X \bullet P]\!]\rho \;\hat{=}\; \text{the unique fixed point of the mapping } M(X,P)\rho.$$

Not all fixed points are unique. For example, every measure is a fixed point of the mapping $M(X,X)\rho$, corresponding to the recursion $\mu X \bullet X$. The rest of this section serves to establish conditions for the existence and uniqueness of fixed points, based on the following theorem:

The Banach Fixed Point Theorem If (M,d) is a complete metric space and $F : M \to M$ is a contraction map, then F has a unique fixed point $fix(F)$. Furthermore, for all S in M, $fix(F) = \lim_{n\to\infty} F^n(S)$. □

For a proof of this theorem see for instance [Su75].

There are two candidates for a suitable metric for PM. The first one takes the weighted sums of all the differences in the probabilities given by the measures P and Q to sets with increasingly longer τ-free prefixes:

$$d(P,Q) \;\hat{=}\; \sum_{n=1}^{\infty} \frac{1}{2^n} \sum_{t\in\Sigma^n} |P\,S(t) - Q\,S(t)|.$$

It is easily checked that this defines a metric. Note that $\forall P, Q \cdot d(P,Q) \leq 2$. The second metric is based on the length of the longest τ-free traces up to which P and Q agree in probability:

$$d'(P,Q) \;\hat{=}\; \inf\{2^{-n} \mid \forall t \in \Sigma^n \cdot P\,S(t) = Q\,S(t)\}.$$

(For both metrics we could have included traces ending in τ in the definition. This would have been topologically equivalent since the probability of any set of traces with a prefix ending in τ is completely determined by the probabilities of the sets of traces with τ-free prefixes. The definition which involves only τ-free traces considerably simplifies the proof of clause 5 (concerning a Lipschitz condition for parallel composition) of lemma 4.2.5 but the other definition would enable us to prove a similar clause for sequential composition. However, since this proof would involve two pages of rather unpleasant algebra and the clause is of minor importance, we have chosen to use the simpler definition.)

In the following we will show (lemma 4.2.2) that convergence in d is equivalent to weak convergence. Convergence in d' implies convergence in d but not vice versa (lemma 4.2.3). This means that d admits a wider variety of recursive definitions. For example, we will be able to deduce (from lemmas 4.2.5 and 4.2.7) that the term $(a \rightarrow X)\ {}_p\!\sqcap X$ corresponds to a contraction map with respect to d but not with respect to d'. Therefore we need to use d. On the other hand, some expressions, like $a \rightarrow (X \| X \| X)$, which we would expect to be well-defined recursions, are not d-contractions. However, they are d'-contractions which together with lemma 4.2.3 is sufficient to assert that a sequence of iterates of this map is also a Cauchy sequence with respect to d. So we also need d'.

Informally, the difference between the two metrics lies in the way they regard probabilistic choice. Take for instance the processes

$$\begin{aligned} P &\triangleq [\![a \rightarrow STOP]\!] \\ Q &\triangleq [\![(a \rightarrow STOP)\ {}_p\!\sqcap\ STOP]\!] \\ R &\triangleq [\![STOP]\!]. \end{aligned}$$

The metric d considers P and Q to be nearer to each other than P and R:

$$\begin{aligned} d(P,Q) &= (1-p)/2 \\ d(P,R) &= 1/2. \end{aligned}$$

The metric d' classes Q and R as equally far apart from P:

$$\begin{aligned} d'(P,Q) &= 1 \\ d'(P,R) &= 1. \end{aligned}$$

Lemma 4.2.2 Convergence in d is equivalent to weak convergence. □

Proof A sequence of probability measures $(P_n)_{n:\mathbb{N}}$ converges in d if and only if it converges on all sets with fixed prefixes.

$$\lim_n d(P_n, P) = 0 \ \Leftrightarrow\ \forall t \in \Sigma^*_+ \cdot \lim_n P_n S(t) = P\,S(t).$$

Since these sets are cylindersets, convergence on all cylindersets implies convergence in d.

For the reverse implication we use the fact that every cylinderset can be written as a countable disjoint union of sets with fixed prefixes. So for any cylinderset A we can write $A = \bigcup_i S_i$ where $\forall i \cdot S_i \in \{S(t) \mid t \in \Sigma_+^*\}$ and $i \neq j \Rightarrow S_i \cap S_j = \emptyset$. Therefore

$$\begin{aligned} \lim_n P_n A &= \lim_n P_n \bigcup_i S_i = \lim_n \sum_i P_n S_i = \sum_i P S_i \\ &= P A. \end{aligned}$$

□

Lemma 4.2.3 Convergence in d' implies convergence in d. □

Proof We show that a Cauchy sequence with respect to d' is also a Cauchy sequence with respect to d. First note that $\forall P, Q \cdot d(P,Q) \leq 2d'(P,Q)$ because

$$d'(P,Q) = \frac{1}{2^n} \Rightarrow \forall t \in \Sigma^n \cdot P S(t) = Q S(t)$$

which means all terms in $d(P,Q)$ involving traces of length less than $n+1$ are zero and

$$d(P,Q) = \sum_{k=n+1}^{\infty} \frac{1}{2^k} \sum_{t \in \Sigma^k} |P S(t) - Q S(t)| \leq \sum_{k=n+1}^{\infty} \frac{1}{2^k} 2 = 2 \frac{1}{2^n}.$$

If $(P_n)_{n:\mathbb{N}}$ is a d'-Cauchy sequence then

$$\forall \epsilon > 0, \exists N, \forall m, n > N \cdot d'(P_n, P_m) < \frac{1}{2}\epsilon$$

which implies $d(P_n, P_m) < \epsilon$. So $(P_n)_{n:\mathbb{N}}$ is also a d-Cauchy sequence. □

Theorem 4.2.4 The space PM is complete in the metric d. □

Proof A metric space is complete if every Cauchy seqnence converges. Let $(P_n)_{n:\mathbb{N}}$ be a Cauchy sequence in PM, that is

$$\forall \epsilon > 0, \exists N, \forall n, m > N \cdot d(P_n, P_m) < \epsilon.$$

If this holds, then $\forall n, m > N$ the difference in probabilities assigned by P_n and P_m to any set $S(t)$ with t of length k can be at most $2^k\epsilon$. Since $2^k\epsilon$ can be made arbitrarily small, $(P_n S(t))_{n:\mathbb{N}}$ is a Cauchy seqnence in $\mathbb{R}$. So we can define a function $Q : \mathcal{F} \to [0,1]$ which assigns to each $S(t)$ the limit $\lim_n P_n S(t)$ and is finitely additive. Then Q is a probability measure on the field of all cylinder sets. By the extension theorem, there exists a unique probability measure P on the σ-field $\mathcal{F}$ which is generated by the cylindersets, snch that P agrees with Q on all cylindersets. But then P is the limit of the Canchy sequence $(P_n)_{n:\mathbb{N}}$. Hence PM is d-complete. □

Now we need to investigate which $PCSP_0$ terms represent contraction maps. A function $F : PM \rightarrow PM$ satisfies a *Lipschitz condition* with constant k if

$$\forall X, Y \in PM \cdot d(F\ X, F\ Y) \leq k\ d(X, Y).$$

Let $r(F)$ denote the smallest such k:

$$r(F) \triangleq \inf \{ k \mid \forall X, Y \in PM \cdot d(F\ X, F\ Y) \leq k\ d(X, Y) \}.$$

The function F is a *contraction map* if $r(F) < 1$, *non-expanding* if $r(F) = 1$ and *expanding* if $r(F) > 1$.

In other models of CSP, prefixing is a contraction map and all other operators except the hiding operators are non-expanding. This is sufficient to turn the composition of any operator with the prefixing operator into a contraction map because the composition of two functions corresponds to the multiplication of their Lipschitz conditions: for any measures X, Y

$$\begin{aligned} d(F(G\ X), F(G\ Y)) &\leq r(F)\ d(G\ X, G\ Y) \\ &\leq r(F)\ r(G)\ d(X, Y) \\ &\leq d(X, Y) \quad \text{if } r(F)\ r(G) < 1. \end{aligned}$$

In the probabilistic model, parallel composition can actually expand the distance between measures, as the following example may illustrate:

Consider the parallel composition of a process with itself, that is let $F \triangleq M(X, X \| X)\rho$. Let

$$\begin{aligned} P &\triangleq [\![a \rightarrow STOP]\!] \\ Q &\triangleq [\![(a \rightarrow STOP)\ {}_p\sqcap\ STOP]\!]. \end{aligned}$$

Then

$$\begin{aligned} F\ P &= [\![X \| X]\!]\rho[P/X] = [\![a \rightarrow STOP]\!] \\ F\ Q &= [\![X \| X]\!]\rho[Q/X] = [\![(a \rightarrow STOP)\ {}_p\sqcap\ STOP]\!] \end{aligned}$$

and $d(P, Q) = (1-p)/2$ whereas $d(F\ P, F\ Q) = (1-p^2)/2$. So

$$d(F\ P, F\ Q) = (1+p)\ d(P, Q).$$

i.e. if p is large parallel composition almost doubles the distance between P and Q.

However, the fact that a function F is an expansion map does not matter as long as $r(F)$ is bounded and can be compensated for by a contraction map G, such that $r(F)\ r(G) < 1$. The following theorem establishes such bounds.

Lemma 4.2.5 Let P, Q be terms possibly involving the term variable Z and let F and G be the corresponding semantic functions, that is let

$$F \triangleq M(Z,P)\rho$$
$$G \triangleq M(Z,Q)\rho.$$

Consider a semantic function H such that

1. H is constant w.r.t. $\rho[\![Z]\!]$. Then $r(H) = 0$.
2. $H = M(Z,Z)\rho$. Then $r(H) = 1$.
3. $H = M(Z, a \to P)\rho$. Then $r(H) = 1/2\ r(F)$.
4. $H = M(Z, P\ {}_p\sqcap\ Q)\rho$. Then $r(H) = p\ r(F) + (1-p)\ r(G)$.
5. $H = M(Z, P \| Q)\rho$. Then $r(H) \leq r(F) + r(G)$.

□

Proof Let X, Y be probability measures. If H is constant then

$$d(H\,X, H\,Y) = 0.$$

If H is the identity function then

$$d(H\ X, H\ \ Y) = d(X,Y).$$

For the third case remember that for $t \in \Sigma^n$, $n > 0$

$$prefix_a^{-1}\ S(t) \ = \ \begin{cases} S(t/1) & \text{if } t_0 = a \\ \emptyset & \text{otherwise .} \end{cases}$$

Therefore

$$\begin{aligned} d(H\,X, H\,Y) &= d([\![a \to P]\!]\rho[X/Z], [\![a \to P]\!]\rho[Y/Z]) \\ &= \sum_{n=1}^{\infty} \frac{1}{2^n} \sum_{t \in \Sigma^n} |F\,X\ prefix_a^{-1} S(t) - F\ Y\ prefix_a^{-1} S(t)| \\ &= \sum_{n=1}^{\infty} \frac{1}{2^n} \sum_{\substack{t/1 \in \Sigma^{n-1} \\ t_0 = a}} |F\,X\ S(t/1) - F\ Y\ S(t/1)| \\ &= \sum_{m=0}^{\infty} \frac{1}{2^{m+1}} \sum_{s \in \Sigma^m} |F\,X\ S(s) - F\ Y\ S(s)| \\ &= \frac{1}{2}\ d(F\,X, F\,Y). \end{aligned}$$

For the last step we could disregard $m = 0$ because $s \in \Sigma^0 \Leftrightarrow s = \langle\rangle$ and $F\,X\ S\langle\rangle = F\ Y\ S\langle\rangle = 0$. For an expression with probabilistic choice the metric d is

$$\begin{aligned} d(H\,X, H\ Y) &= \sum_{n=1}^{\infty} \frac{1}{2^n} \sum_{t \in \Sigma^n} |p\,F\,X\ S(t) + (1-p)\,G\,X\ S(t) \\ &\qquad - p\,F\,Y\ S(t) - (1-p)\,G\,Y\ S(t)| \\ &\leq p\ d(F\,X, F\ Y) + (1-p)\ d(G\,X, G\ Y). \end{aligned}$$

To show that parallel composition has Lipschitz condition at most the sum of the Lipschitz conditions of its components is slightly more involved. Note that for a τ-free trace t the inverse image $par^{-1}S(t)$ is simply $S(t) \times S(t)$. So if H is the semantic function corresponding to the parallel composition $P \| Q$ of the $PCSP_0$-terms P and Q we get

$$d(H\ X, H\ Y) = \sum_{n=1}^{\infty} \frac{1}{2^n} \sum_{t \in \Sigma^n} |F\,X\,S(t)\ G\,X\,S(t) - F\,Y\,S(t)\ G\,Y\,S(t)|.$$

The terms in this summation are of the form $|a_1 b_1 - a_2 b_2|$ where a_1, a_2, b_1 and b_2 are probabilities. Writing

$$a_1 b_1 - a_2 b_2 = \frac{1}{2}(a_1 - a_2)(b_1 + b_2) + \frac{1}{2}(a_1 + a_2)(b_1 - b_2)$$

we can derive the inequality

$$\begin{aligned} |a_1 b_1 - a_2 b_2| &\leq \frac{1}{2}\,|\,|a_1 - a_2|(b_1 + b_2) + (a_1 + a_2)|b_1 - b_2|\,| \\ &\leq |a_1 - a_2| + |b_1 - b_2|. \end{aligned} \tag{4.1}$$

The last step follows because $b_1 + b_2 \leq 2$ and $a_1 + a_2 \leq 2$. Using this inequality we can split $d(H\,X, H\ Y)$ into two sums, one involving only terms in F and the other only involving terms in G. This leads to

$$\begin{aligned} & d(H\,X, H\ Y) \\ &< \sum_{n=1}^{\infty} \frac{1}{2^n} \sum_{t \in \Sigma^n} (|F\,X\,S(t) - F\,Y\,S(t)| + |G\,X\,S(t) + G\,Y\,S(t)|) \\ &\leq d(F\,X, F\ Y) + d(G\,X, G\ Y) \\ &\leq r(F)\ d(X, Y) + r(G)\ d(X, Y). \end{aligned}$$

So $r(H) < r(F) + r(G)$.

□

Lemma 4.2.5 provides a simple rule to determine whether a $PCSP_0$-term is a contraction map w.r.t. d. It shows that, unlike in other metrically based *CSP* models, unguarded recursion may sometimes be well-defined in the probabilistic model. For example,

$$\begin{aligned} r((a \to (X \| STOP)) \,_p\!\sqcap (X \| X)) &< p\,\frac{1}{2}(1+0) + (1-p)\,2 \\ &< 1 \qquad \text{if } p > \frac{2}{3}. \end{aligned}$$

However, some expressions, like $a \to (X \| X \| X)$, which we would expect to provide well-defined recursions, are not contraction maps with respect to d. The next lemma shows that all guarded recursions are contraction maps with respect to d'. Thus the fixed-point theorem applied to (PM, d') together with lemma 4.2.3 ensure that any guarded recursion is well-defined.

The metric d' is analogous to the metrics which have been used for the untimed and timed models of *CSP* (for a summary cf. [Re88]) in that it depends solely on the number of steps up to which the behaviour of two processes is indistinguishable. Not surprisingly it is also an an ultra-metric (i.e. $\forall X, Y, Z \in PM \cdot d'(X,Y) \leq \max(d'(X,Y), d'(Y,Z))$). In the non-probabilistic models a function is a contraction map if and only if it increases the number of steps up to which the behaviour of two processes is indistinguishable. This is also true of functions which are d'-contractions. We therefore adopt the standard terminology for such functions ([Ros82], [Hoa85]):

Definition 4.2.6 Let P be a $PCSP_0$-term possibly involving a free variable Z. We say that P is *constructive* if $M(Z,P)\rho$ is a d'-contraction, and *non-destructive* if $M(Z,P)\rho$ is non-expanding with respect to d'. □

So in a probabilistic context a function is constructive if and only if it increases the length of the traces up to which two processes agree in probability:

$$\begin{aligned} P \text{ is constructive} &\Leftrightarrow d'([\![P]\!]\rho[X/Z], [\![P]\!]\rho[Y/Z]) < d'(X,Y) \\ &\Leftrightarrow (\forall t \in \Sigma^n \cdot X\,S(t) = Y\,S(t) \\ &\qquad \Rightarrow \forall s \in \Sigma^{n+1} \cdot [\![P]\!]\rho[X/Z]\,S(s) = [\![P]\!]\rho[Y/Z]\,S(s)). \end{aligned}$$

Similarly for non-destructive terms.

Lemma 4.2.7

1. *STOP* and *SKIP* are constructive.
2. The free variable X is non-destructive.
3. $a \rightarrow P$ is constructive if P is non-destructive.
4. $P \,{}_p\!\sqcap\, Q$, $P \| Q$ and $P\,;Q$ are constructive if P and Q are constructive.

□

Proof Let X, Y be two measures and suppose that $\forall t \in \Sigma^n \cdot X\ S(t) = Y\ S(t)$. Let $s \in \Sigma^{n+1}$. Clauses 1 and 2 follow directly. Clause 3 follows because $prefix_a^{-1} S(s)$ gives a set with a fixed prefix of length n:

$$\begin{aligned}
[\![a \rightarrow P]\!]\rho[X/Z]\,S(s) &= \begin{cases} [\![P]\!]\rho[X/Z]\,S(s/1) & \text{if } s_0 = a \\ 0 & \text{otherwise} \end{cases} \\
&= \begin{cases} [\![P]\!]\rho[Y/Z]\,S(s/1) & \text{if } s_0 = a \\ 0 & \text{otherwise} \end{cases} \\
&\quad \text{since } s/1 \in \Sigma^n \\
&= [\![a \rightarrow P]\!]\rho[Y/Z]\,S(s).
\end{aligned}$$

Let $s \in \Sigma^n$. Probabilistic choice is non-destructive because it does not affect the $S(s)$:

$$\begin{aligned}
[\![P \,{}_p\!\sqcap\, Q]\!]\rho[X/Z]\,S(s) &= p[\![P]\!]\rho[X/Z]\,S(s) + (1-p)[\![Q]\!]\rho[X/Z]\,S(s) \\
&= p[\![P]\!]\rho[Y/Z]\,S(s) + (1-p)[\![Q]\!]\rho[Y/Z]\,S(s) \\
&= [\![P \,{}_p\!\sqcap\, Q]\!]\rho[Y/Z]S(s).
\end{aligned}$$

For parallel composition recall that d' only compares τ-free s. For these

$$\begin{aligned}
[\![P \| Q]\!]\rho[X/Z]\,S(s) &= [\![P]\!]\rho[X/Z]\,S(s)\ [\![Q]\!]\rho[X/Z]\,S(s) \\
&= [\![P]\!]\rho[Y/Z]\,S(s)\ [\![Q]\!]\rho[Y/Z]\,S(s) \\
&= [\![P \| Q]\!]\rho[Y/Z]\,S(s).
\end{aligned}$$

For the proof for sequential composition assume that s $\surd$- and τ-free. Then

$$\begin{aligned}
&[\![P\,;Q]\!]\rho[X/Z]\,S(s) \\
&= \sum_{k=0}^{n-1} [\![P]\!]\rho[X/Z]\,S(s \restriction k\langle\surd\rangle)\ [\![Q]\!]\rho[X/Z]\,S(s/k) + [\![P]\!]\rho[X/Z]\,S(s) \\
&= \sum_{k=0}^{n-1} [\![P]\!]\rho[Y/Z]\,S(s \restriction k\langle\surd\rangle)\ [\![Q]\!]\rho[Y/Z]\,S(s/k) + [\![P]\!]\rho[Y/Z]\,S(s) \\
&= [\![P\,;Q]\!]\rho[Y/Z]\,S(s).
\end{aligned}$$

Similarly for traces ending in $\surd$. □

We combine lemmas 4.2.5 and 4.2.7 to characterise a class of recursive expressions which are well-defined.

Theorem 4.2.8 Suppose that P is a $PCSP_0$ expression possibly containing the free variable X. If $r(M(X,P)\rho) < 1$ or if P is constructive with respect to X then the semantics

$$[\![\mu X \bullet P]\!]\rho$$

is well defined for all bindings ρ. □

For well-defined $\mu X \bullet P$ the laws listed below apply. The first two, concerning the unfolding of recursion and the changing of bound variables, are completely standard and follow directly from the semantics. The last one is particular to the probabilistic model and follows from the last theorem.

Lemma 4.2.9

L1 $\mu X \bullet P \equiv P[\mu X \bullet P/X]$.

L2 If Y is not free in P then $\mu X \bullet P \equiv \mu Y \bullet P$.

L3 If $M(X, P)\rho$ is a contraction map w.r.t. d then $\mu X \bullet (P \,{}_p\!\sqcap\, X) \equiv \mu X \bullet P$.

□

Law 1 justifies the use of recursive equations as process definitions. Since $P \equiv Q[P/X]$ if and only if $P \equiv \mu X \bullet Q$ we write $P = Q[P/X]$ as an alternative to $P \triangleq \mu X \bullet Q$. So for example $P = a \rightarrow P$ and $P \triangleq \mu X \bullet a \rightarrow X$ are equivalent definitions. The equational definition is more concise, especially in the case of mutual recursion.

4.3 Mutual Recursion

To give a semantics to mutual recursion we closely follow the approach which [DS91] presented for timed *CSP*. We use the same syntax and translate the semantics from the domain of sets of timed traces to the domain of probability measures.

A term P may be defined by a vector of mutually recursive equations with an initial index $j \in I$ to indicate the starting point of the recursion:

$$P \triangleq \langle X_i = P_i \rangle_j \qquad i \in I.$$

Each term P_i may contain calls to any of the variables X_j. The index set I need not be finite.

As an example, consider the process-algebra representation of a random walk on the natural numbers: the walk starts off at the origin:

$$CT = CT_0.$$

At the origin it either goes up or it stays at the origin. At any other point it either goes up or down. Any alternative is chosen with probability 1/2.

$$\begin{aligned} CT_0 &= around \rightarrow CT_0 \;{}_{\frac{1}{2}}\!\sqcap\; up \rightarrow CT_1 \\ CT_n &= down \rightarrow CT_{n-1} \;{}_{\frac{1}{2}}\!\sqcap\; up \rightarrow CT_{n+1} \qquad n > 0. \end{aligned}$$

The semantic domain required to model a solution for a vector of mutually recursive equations is PM^I; this is a product space with one copy of the model

PM for each $i \in I$. For any I, this domain is a complete metric space, with the following metric on vectors.

$$\boldsymbol{d}(\boldsymbol{V}, \boldsymbol{W}) \;\hat{=}\; \sup \{d(V_i, W_i) \mid i \in I\}.$$

To construct a semantic function for vectors of terms, we extend the use of bindings to include mappings from vectors of variables to vectors of processes. We overload the mapping notation (definition 4.2.1) with

Definition 4.3.1 If $\boldsymbol{P}$ a vector of $PCSP_0$ terms, and $\boldsymbol{X}$ is a vector of variables indexed by the set I, then

$$M(\boldsymbol{X}, \boldsymbol{P})\rho \;\hat{=}\; \lambda \boldsymbol{Y} \cdot [\![\boldsymbol{P}]\!]\rho[\boldsymbol{Y}/\boldsymbol{X}].$$

is the mapping corresponding to the semantics of $\boldsymbol{P}$ as a function of the processes bound to $\boldsymbol{X}$. □

Definition 4.3.2 If $\boldsymbol{P}$ is a vector of $PCSP_0$ terms, then

$$[\![\langle X_i = P_i \rangle_j]\!]\rho \;\hat{=}\; S_j \text{ where } \boldsymbol{S} \text{ is a fixed point of } M(\boldsymbol{X}, \boldsymbol{P})\rho.$$

□

This semantics is well-defined if all fixed points of the mapping $\boldsymbol{M}(\boldsymbol{X}, \boldsymbol{P})\rho$ agree on the j component, which is trivially the case if $M(\boldsymbol{X}, \boldsymbol{P})\rho$ has only one fixed point. For this to be true, it is sufficient that every P_i be a contraction mapping for every X_j with respect to d, as this turns $\boldsymbol{P}$ into a contraction mapping for $\boldsymbol{X}$ with respect to $\boldsymbol{d}$. It is also sufficient if every P_i is constructive for every X_j, as this turns $\boldsymbol{P}$ into a contraction mapping for $\boldsymbol{X}$ with respect to $\boldsymbol{d}'$, which implies convergence with respect to $\boldsymbol{d}$. [DS91] show that this condition can be weakened in the following way.

A partial ordering $\prec$ on a set I is a well-ordering if and only if there is no infinite strictly descending sequence $(s_i)_{i:\mathbb{N}}$ such that $\forall\, i : \mathbb{N} \bullet s_{i+1} \prec s_i$.

Definition 4.3.3 If $\prec$ is a partial ordering on I, and i is an element of I, then the initial segment of i in $(I, \prec)$ is defined by $\mathrm{seg}(i) \;\hat{=}\; \{j : I \mid j \prec i\}$. □

Definition 4.3.4 A vector of terms $\boldsymbol{P}$ is constructive for a vector of variables $\boldsymbol{X}$ if there is a well-ordering $\prec$ of the indexing set I such that

$$\forall j, i : I \cdot j \notin \mathrm{seg}(i) \;\Rightarrow\; P_i \text{ is constructive for } X_j$$
$$\forall j, i : I \cdot j \in \mathrm{seg}(i) \;\Rightarrow\; P_i \text{ is non-destructive for } X_j.$$

□

Any mutual recursion in which the vector of terms is constructive for the vector of variables has a well-defined semantics.

Theorem 4.3.5 (Unique Fixed Point Theorem) If a vector of terms $\boldsymbol{P}$ is constructive for the vector of variables $\boldsymbol{X}$, then the mapping $M(\boldsymbol{X}, \boldsymbol{P})\rho$ has a unique fixed point in PM^I. □

The proof of this theorem is given in [Dav91]. From it we deduce the corollary:

Corollary 4.3.6 If a vector of terms $\boldsymbol{P}$ is constructive for vector of variables $\boldsymbol{X}$, then the recursion $\mu\, \boldsymbol{X} \bullet \boldsymbol{P}$ is well-defined. □

4.4 Recursion Induction

The bare-hands approach to proving that a recursively defined process P has a property R involves three proof obligations:

1. R is a satisfiable predicate (that is $\exists\, P \cdot R(P)$),
2. R is continuous
   (so if (P_i) is a convergent sequence then $\forall\, i \cdot R(P_i) \Rightarrow R(\lim\ P_i)$),
3. There exists a convergent sequence (P_i) such that $P_i \xrightarrow{w} P$ and $\forall\, i \cdot R(P_i)$.

The theory of recursion induction, as presented by [Ros82] and extended to timed *CSP* by [Re88] and [DS91], simplifies these obligations by establishing

* a criterion for the continuity of a predicate which is easily checked,
* an inference rule which reduces the third obligation to one step.

We apply this approach to the probabilistic model.

We identify predicates on measures with mappings from PM to the space of truth values $TV \triangleq \{true, false\}$. We use the metric d' to define the open sets of PM to be those generated by the open balls, and define the open sets of TV as $\{\emptyset, \{false\}, \{true, false\}\}$ (this is the Sierpinsky topology). We now show that a predicate is continuous if we only need to look at sets with fixed prefixes to establish whether it holds of a process.

Theorem 4.4.1 If R is a mapping from the complete metric space (PM, d') to TV such that for any P in PM

$$R(P) = false \;\Rightarrow\; (\exists\, n : \mathbb{N} \cdot \forall\, t \in \Sigma^n \cdot P\,S(t) = P'S(t) \Rightarrow R(P') = false)$$

then R is continuous. □

Proof A mapping between two topological spaces is continuous if the inverse image of every open set is open. Recall that the metric d' depends on the length of the longest prefix up to which P and P' agree in probability.

$$d'(P,Q) \;=\; \inf\{2^{-n} \mid \forall t \in \Sigma^* \cdot \#t \leq n \Rightarrow P\ S(t) = Q\ S(t)\}.$$

The above condition implies that whenever $R(P)$ is false it is false of all measures in the open ball $\{P' \mid d(P,P') < 2^{n+1}\}$. Thus R is continuous. □

The following theorem is taken from [Ros82].

Theorem 4.4.2 Let $M = (A,d)$ be a complete metric space, and let TV be the topological space $(\{true, false\}, \mathcal{T})$ with the Sierpinsky topology. If $F : M \to TV$ is continuous and the set $\{a \in A \mid F(a) = true\}$ is nonempty, then

$$(\forall x : A \cdot F(x) = true \Rightarrow F(C(x)) = true) \;\Rightarrow\; F(fix(C)) = true$$

for any contraction mapping $C : M \to M$. □

This allows us to postulate the following inference rule.

Suppose that R is a satisfiable and continuous predicate and that the $PCSP_0$-term P is constructive for the variable X. Then

Rule 4.4.3

$$\frac{\forall Y : PM \cdot R(Y) \Rightarrow R([\![P]\!]\rho[Y/X])}{R([\![\mu X \cdot P]\!]\rho)}$$

□

Proof If P is constructive then $\lambda Y \cdot [\![P]\!]\rho[Y/X]$ is a contraction mapping. We have assumed that R is satisfiable, continuous and that $\forall Y : PM \cdot R(Y) \Rightarrow R([\![P]\!]\rho[Y/X])$. Therefore by theorem 4.4.2 the rule is sound. □

This rule can be extended to mutually recursive equations as shown by [DS91]. If $\boldsymbol{P}$ is a vector of mutually recursive processes which is constructive for the vector of variables $\boldsymbol{X}$ then to establish that a vector of predicates $\boldsymbol{R}$ correctly describes the fixed point of $M(\boldsymbol{X}, \boldsymbol{P})$ it is sufficient to show that each R_i is continuous and satisfiable and that $\boldsymbol{R}$ is preserved by $M(\boldsymbol{X}, \boldsymbol{P})$.

Rule 4.4.4

$$\frac{(\forall i \cdot R_i(Y_i)) \Rightarrow \forall j \cdot R_j([\![P_j]\!]\rho[\boldsymbol{Y}/\boldsymbol{X}])}{\boldsymbol{R}([\![\mu \boldsymbol{X} \cdot \boldsymbol{P}]\!]\rho)}$$

□

Chapter 5

Examples

At the beginning of this thesis we claimed that the specification of probabilistic processes must be linked to a notion of probabilistic correctness which requires that a property be satisfied with probability 1. In this chapter we give examples of some typical properties of probabilistic processes and show how the semantics of $PCSP_0$ enable us to reason about them.

A property which holds of all traces except possibly a set of traces of zero probability is said to hold of *almost all* traces.

5.1 Fairness

The first property we consider is *fairness*: if a probabilistic choice of finitely many branches is executed infinitely often then almost all traces contain every branch infinitely often: the probability that from some point onwards one branch is overlooked forever is zero. This coincides with the notion of *extreme fairness* introduced by [Pnu83]. The probability that a process is fair can be evaluated with the help of the following lemma, which we quote from [Bi79].

The Second Borel-Cantelli Lemma If (A_n) is a sequence of independent events and $\sum_n P\ A_n$ diverges then $P(\limsup_n A_n) = 1$. □

(Note that "event" here means "a set of points in a probability space", not to be confused with "event" as a synonym for action.)

Lemma 5.1.1 Let

$$P \ = \ \prod_{p_n}(a_n \rightarrow P) \quad \text{where } 0 \leq n < N, \text{ the } a_n \text{ are distinct and } p_n > 0.$$

Then P is fair in the sense that almost all traces of P contain every a_n infinitely often, i.e.

$$\forall 0 \leq n < N \cdot [\![P]\!] \limsup_i \{u \mid u_i = a_n\} = 1.$$

□

Proof Let A_i be the set of traces whose $(i+1)^{\text{th}}$ element is a_n.

$$A_i \;\hat{=}\; \{u \mid u_i = a_n\} \;=\; \bigcup_{s \in \Sigma^i} S(s\langle a_n \rangle).$$

For all $s \in \Sigma^i$ we have $[\![P]\!] S(s\langle a_n \rangle) = p_n [\![P]\!] S(s)$. Also $\sum_{s \in \Sigma^i} [\![P]\!] S(s) = 1$. Hence

$$[\![P]\!] A_i \;=\; \sum_{s \in \Sigma^i} [\![P]\!] S(s\langle a_n \rangle) \;=\; p_n.$$

Similarly, it can be shown that if $i \neq j$ then any A_i, A_j are independent, i.e. $P(A_i \mid A_j) = p_n$. Consider the set

$$\limsup_i A_i \;=\; \bigcap_{i=1}^{\infty} \bigcup_{k=i}^{\infty} A_k$$

consisting of all the traces which contain a_n infinitely often. The A_i are independent and the sum $\sum_i [\![P]\!] A_i = \sum_i p_n$ diverges. Therefore by the Borel-Cantelli lemma $[\![P]\!](\limsup_i A_i) = 1$. Hence P is fair. □

This result can easily be generalised: Suppose $P' = \prod_{p_n} P_n$ where each P_n contains only prefixing and a recursive invocation of P'. Let $t(P_n)$ be the trace which P_n performs in one unfolding of the recursion and let f be a function such that $\forall n \cdot f(a_n) = t(P_n)$. Lifting f to sequences we can write $\forall A \cdot [\![P']\!] A = [\![P]\!] f^{-1} A$. In particular, the set of traces which contain $t(P_n)$ infinitely often has inverse image $(\limsup_i A_i)$ where A_i as defined above. Hence it also has probability 1.

As a concrete example of a process of this form consider a communications medium which loses input with probability p. Since we are not interested in the nature of the data which is being transmitted, we model this simply as a process which can perform two actions: *in* and *out*, such that the probability of an *in* being followed by *out* is $1-p$.

$$P \;=\; in \rightarrow (P \;{}_p\!\sqcap\; out \rightarrow P).$$

Since probabilistic choice distributes through prefixing we can write

$$P \;=\; (in \rightarrow P) \;{}_p\!\sqcap\; (in \rightarrow out \rightarrow P).$$

From the generalisation of lemma 5.1.1 it follows that P will perform (in, out) infinitely often with probability 1. In other words we know that P will never stop producing output altogether – if this was not true, it would be impossible to implement a working communications protocol around P.

5.2 The Asymptotic Frequency of an Action

When we defined the interleaving operator we claimed that the asymptotic frequency of heads in the traces of a coin-tossing process is the same as the probability with which a head appears at each throw. This example shows how to substantiate this claim.

An action a occurs with *asymptotic frequency* l in an infinite trace u if the ratio of occurrences of a to the length of successively longer prefixes of u tends towards the limit l:

$$\lim_{n\to\infty} \frac{(u\lceil n)\downarrow\{a\}}{n} \;=\; l.$$

Note that this limit does not exist for every trace. A counterexample is provided by the trace $(a, b, b, a, a, a, a, \ldots)$ in which each run of a's is followed by twice as many b's and vice versa.

We say that the process P performs a with asymptotic frequency l if the probability of the traces in which a occurs with asymptotic frequency l is 1:

$$P\,\{u \mid \lim_{n\to\infty} \frac{(u\lceil n)\downarrow\{a\}}{n} = l\} \;=\; 1.$$

The set of these traces is measurable because it can be expressed in terms of cylinder sets as follows:

$$\begin{aligned}
&\{u \mid \lim_{n\to\infty} \frac{(u\lceil n)\downarrow\{a\}}{n} = l\} \\
&\quad= \{u \mid \forall \epsilon > 0 \cdot \exists N \cdot \forall n > N \cdot \left|\frac{(u\lceil n)\downarrow\{a\}}{n} - l\right| < \epsilon\} \\
&\quad= \bigcap_{m=1}^{\infty} A_m
\end{aligned}$$

where

$$A_m \;=\; \bigcup_{N=0}^{\infty} \bigcap_{n>N} \{u \mid \left|\frac{(u\lceil n)\downarrow\{a\}}{n} - l\right| < \frac{1}{m}\}.$$

In the following lemma we show that the processes which we considered in the previous section not only perform every branch infinitely often, but with a constant asymptotic frequency:

Lemma 5.2.1 Let

$$P \;=\; \textstyle\bigsqcap_{p_n}(a_n \to P) \quad \text{where } 0 \le n < N, \text{ the } a_n \text{ are distinct and } p_n > 0.$$

Then P performs each a_n with asymptotic frequency p_n. □

Proof Consider the set $R(i,j)$ of traces which contain a run of j actions other than a_n after the i^{th} a_n.

$$\begin{aligned} R(0,j) &\triangleq \{u \mid (u\restriction j)\downarrow\{a_n\} = 0 \wedge u_j = a_n\} \\ R(i,j) &\triangleq \{t\langle a_n\rangle s\langle a_n\rangle u \mid \#s = j \wedge s\downarrow\{a_n\} = 0 \wedge t\downarrow\{a_n\} = i-1\} \quad i > 0. \end{aligned}$$

Then $[\![P]\!]\, R(0,j) = p_n(1-p_n)^j$ and for $i > 0$

$$\begin{aligned} [\![P]\!]\, R(i,j) &= \sum_{k=0}^{\infty} \binom{i+k-1}{k} (1-p_n)^i p_n^k\, (1-p_n)^j p_n \\ &= (1-p_n)^j p_n. \end{aligned}$$

Thus $[\![P]\!]\, R(i,j)$ is independent of i. Similarly it can be shown that the probabilities of two different runs are independent of each other. Let V_i be a random variable which records the number of actions other than a_n in the i^{th} run, that is $V_i(u) = \sum_j j\ I_{R(i,j)}(u)$. The sequence (V_i) is a sequence of independent, identically distributed random variables, each of which has expected valne

$$E(V) = \sum_{j=0}^{\infty} j(1-p_n)^j p_n = \frac{(1-p_n)}{p_n}.$$

This translates into an expected ratio of the number of a_n's to the length of each run of $1/(E(V)+1) = p_n$. The strong law of large numbers (cf. [Bi79]) applies to give

$$[\![P]\!]\{u \mid \lim_{i\to\infty} \frac{V_1 + \ldots + V_i}{i}(u) = E(V)\} = 1,$$

i.e. the asymptotic ratio of a_n's in all runs is the same as the expected ratio in each run. Now

$$\begin{aligned} &\frac{V_1 + \ldots + V_i}{i}(u) = x \\ &\Leftrightarrow \exists j_1, j_2, \ldots j_i \cdot u \in R(1,j_1) \cap R(2,j_2) \cap \ldots \cap R(i,j_i) \\ &\qquad \wedge \frac{(u\restriction \sum_k j_k + i)\downarrow\{a_n\}^c}{i} = x \\ &\qquad \wedge \frac{(u\restriction \sum_k j_k + i)\downarrow\{a_n\}}{i} = 1 \\ &\qquad \wedge \sum_k j_k = i\,x \end{aligned}$$

So

$$\lim_{i\to\infty} \frac{V_1 + \ldots + V_i}{i}(u) = E(V)$$

$$\Rightarrow \lim_{i\to\infty} \frac{(u\restriction i(E(V)+1)))\downarrow\{a_n\}}{i} = 1$$

$$\Rightarrow \lim_{i\to\infty} \frac{(u\restriction i(E(V)+1)))\downarrow\{a_n\}}{i(E(V)+1)} = \frac{1}{(E(V)+1)}$$

$$\Rightarrow \lim_{k\to\infty} \frac{(u\restriction k)\downarrow\{a_n\}}{k} = p_n$$

Hence as required

$$[\![P]\!]\{u \mid \lim_{i\to\infty} \frac{(u\restriction i)\downarrow\{a_n\}}{i} = p_n\} \;=\; 1.$$

□

As in the preceeding section, this lemma can be generalised to processes of the form $P' = \prod_{p_n} P_n$ where P_n contains only prefixing and a recursive invocation of P'. Since the asymptotic frequency of each trace $t(P_n)$ is p_n it follows that the asymptotic frequency of any single action of P' is $\sum_n (p_n/\#t(P_n))$.

5.3 Deadlock

In this section we show that if one of the branches of a probabilistic choice ends in deadlock, then a repeated execution of this choice will eventually deadlock with probability 1.

Lemma 5.3.1 Let $P = \prod_{p_n} P_n$ where $p_0 > 0$, $P_0 = STOP$ and for all $n > 0$, P_n contains only prefixing and a recursive invocation of P. Then P will deadlock with probability 1. □

Proof The set D of all traces after which P may deadlock can be written as

$$D \;=\; \bigcup_{k=0}^{\infty} A_k$$

where $A_k = \bigcup_{t\in\Sigma^k} S(t\langle\tau\rangle)$ denotes the set of traces such that P deadlocks after k steps. Then

$$[\![P]\!]\, A_k \;=\; (1-p_0)^k\, p_0$$

and

$$[\![P]\!]\, D \;=\; \sum_{k=0}^{\infty} [\![P]\!]\, A_k \;=\; 1.$$

□

We can use this lemma to prove, for example, that two coin-tossing processes will eventually deadlock when put in parallel: let $P = hd \rightarrow P \ {}_p\sqcap \ tl \rightarrow P$ and $Q = hd \rightarrow Q \ {}_q\sqcap \ tl \rightarrow Q$. From the laws of parallel composition it follows that

$$P \parallel Q \ \equiv \ hd \rightarrow (P \parallel Q) \ {}_s\sqcap \ (tl \rightarrow (P \parallel Q) \ {}_r\sqcap \ STOP)$$
$$\text{where } s = pq, \ r = (1-p)(1-q)/(1-pq).$$

This is of the form described by lemma 5.3.1. The result follows.

5.4 Random Walk on the Positive Integers

In [Hoa85], p.174, the example of a counter which can move *up* or *around* at ground-level and *up* or *down* above ground-level is used to show how different algebraic representations of a process can be proved equivalent algebraically. If instead of deterministic choice we use probabilistic choice with probability $1/2$ the process becomes a random walk on the natural numbers: let $C \triangleq CT_0$ where

$$CT_0 \ = \ around \rightarrow CT_0 \ {}_{\frac{1}{2}}\sqcap \ up \rightarrow CT_1$$
$$CT_n \ = \ down \rightarrow CT_{n-1} \ {}_{\frac{1}{2}}\sqcap \ up \rightarrow CT_{n+1}, \qquad n > 0.$$

For an alternative representation of this process, take

$$ZERO \ = \ around \rightarrow ZERO \ {}_{\frac{1}{2}}\sqcap \ up \rightarrow POS \, ; ZERO$$
$$POS \ = \ down \rightarrow SKIP \ {}_{\frac{1}{2}}\sqcap \ up \rightarrow POS \, ; POS.$$

and put

$$C_0 \ = \ ZERO, \ C_{n+1} \ = \ POS \, ; C_n.$$

The proof that $\forall n. C_n = CT_n$ follows along the same lines as for the original example (cf. [Ros82]) because all the relevant laws for deterministic choice are replaced by corresponding laws for probabilistic choice. Instead of repeating it here we prove that the counter eventually returns to zero with probability 1. The probability of eventual return to zero is the sum of the probabilities of the first return to zero occurring after n steps, which we denote by r_n. Clearly, the probability of first return to zero at step 1 is the probability of staying at zero at step 1, i.e.

$$r_1 \ = \ [\![C]\!] \, S(around) \ = \ \frac{1}{2}.$$

If the first return to zero happens at some later stage there must have been the same number of *up*'s as *down*'s and no *around*. Moreover, up until the last step there must always have been more *up*'s than *down*'s. So the probability of first return to zero after an odd number of steps is

$$r_{2n+1} = 0$$

and the probability of first return to zero after an even number of steps is

$$r_{2n} = \sum_t [\![C]\!] S(t) \quad \text{where } \#t = 2n \wedge t \downarrow up = t \downarrow down \wedge t \downarrow around = 0$$
$$\wedge \forall k < 2n \cdot (t \restriction k) \downarrow up > (t \restriction k) \downarrow down.$$

Every set $S(t)$ where t is a trace of length $2n$ bas probability $(1/2)^{2n}$ and r_{2n} is this number multiplied by the number of traces t which satisfy the above constraints.

The number of traces can be determined by the following standard approach (cf. [Fel57]): let s_k represent the difference between the number of *up*'s and *down*'s after the first k steps. So $s_0 = 0$ and $s_k - s_{k-1} = \pm 1$. We represent a sequence of *up*'s and *down*'s by a polygonal line whose vertices have abscissas $0, 1, \ldots k$ and ordinates $s_0, s_1 \ldots s_k$. Such lines are called paths.

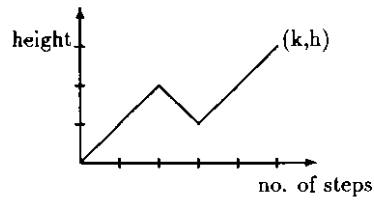

Figure 5.1: A path

A path from the origin to an arbitrary point (k, h) exists only if $k = i + j$ and $h = i - j$ where i, j are the nnmhers of *up*'s and *down*'s respectively. In this case there are

$$N_{k,h} = \binom{i+j}{i} = \binom{k}{(k+h)/2}$$

ways of getting from the origin to (k, h). If the starting point is (k_1, h_1) and the end point (k_2, h_2) there are $N_{k_2-k_1, h_2-h_1}$ ways of getting from one to the other.

However, we are only interested in those paths which do not cross the x-axis. Their number can be determined with the help of the reflection principle, which we quote from [Fel57]:

The Reflection principle Let $A = (k_a, h_a)$ and $B = (k_b, h_b)$ be two integral points such that $k_b > k_a \geq 0$ and $h_b > 0, h_a > 0$. By reflection on the x-axis is meant the point $A' = (k_a, -h_a)$. The number of paths from A to B which touch or cross the x-axis equals the number of paths from A' to B. □

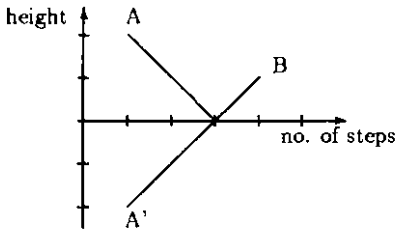

Figure 5.2: The reflection principle

As an immediate consequence the number of paths from A to B which do *not* touch the x-axis can be calculated as the total number of paths from A to B minus the number of paths from A' to B.

The paths which start at the origin and first return to it at step $2n$ must all pass through the points $(1,1)$ and $(2n-1,1)$. Between these two points the number of paths which do not touch the x-axis is

$$N_{2n-2,0} - N_{2n-2,2} = \binom{2n-2}{n-1} - \binom{2n-2}{n} = \frac{1}{n}\binom{2n-2}{n-1}.$$

So

$$r_{2n} = \frac{1}{n}\binom{2n-2}{n-1}\left(\frac{1}{2}\right)^{2n}.$$

To sum all the return probabilities we factorise this expression into

$$r_{2n} = \frac{1}{2}(u_{2n-2} - u_{2n}) \quad \text{where } u_{2n} = \binom{2n}{n} 2^{-2n}.$$

(It can be shown that $1/2\ u_{2n}$ is the probability of no return in the first $2n$ steps. So r_{2n} is the probability of no return up to $2n-2$ steps minus the probability that the process still hasn't returned to zero at $2n$ steps.) It follows that

$$r_1 + \sum_{n=1}^{\infty} r_{2n} = \frac{1}{2} + \frac{1}{2}u_0 = 1.$$

Hence the process is certain to return to zero eventually.

5.5 An Interesting Fixed Point

This example stems from [Ros88] who used it to demonstrate that the failures-divergences model witb infinite traces can cope witb unbounded non-determinism, but that transfinite induction may be needed to compute the fixed point of a recursion in this model. Replacing non-deterministic choice by probabilistic choice gives a recursive process definition in the probabilistic model whose fixed point can be computed by ordinary induction, but whose form is still somewhat surprising. Let Q be a process which offers an unbounded probabilistic choice of performing some finite number of a's:

$$Q \;\hat{=}\; \prod_{p_n} Q_n \quad \text{where } n \geq 0,\ Q_0 \hat{=} \mathit{STOP} \text{ and } Q_{n+1} \hat{=} a \rightarrow Q_n.$$

Let

$$P \;=\; (a \rightarrow P) \parallel Q.$$

We show that

$$P \;\equiv\; \prod_{q_n} Q_n \quad \text{where } Q_n \text{ as above, } q_0 = p_0, \text{ and } q_{n+1} = \prod_{j=0}^{n}(1 - \sum_{k=0}^{j} p_k) \sum_{k=0}^{n+1} p_k.$$

The predicate $R(Y) \;\hat{=}\; (Y = [\![\prod_{q_n} Q_n]\!])$ is satisfiable and continuous and the term $(a \rightarrow X) \parallel Q$ is constructive for the variable X. We can therefore apply the inference rule for recnrsion induction (4.4.3). Its antecedent requires that

$$\forall\, Y : PM \cdot R(Y) \Rightarrow R([\![a \rightarrow X \parallel Q]\!]\rho[Y/X]).$$

Suppose that $Y = [\![\prod_{q_n} Q_n]\!]$. Substituting for X in $a \rightarrow X \parallel Q$ gives

$$\begin{aligned}
(a \rightarrow \prod_{q_n} Q_n) \parallel Q \;&\equiv\; (\prod_{q_n} a \rightarrow Q_n) \parallel (\prod_{p_k} Q_k) \\
&\quad \text{since } \rightarrow \text{ distributes over } \sqcap \\
&\equiv\; \prod_{q_n} (Q_{n+1} \parallel \prod_{p_k} Q_k) \\
&\quad \text{since } a \rightarrow Q_n = Q_{n+1} \text{ and } \sqcap \text{ distributes over } \parallel \\
&\equiv\; \prod_{q_n} \prod_{p_k} (Q_{n+1} \parallel Q_k) \\
&\quad \text{since } \sqcap \text{ distributes over } \parallel \text{ and is associative.}
\end{aligned}$$

From the laws which relate prefixing and parallel composition it follows that if $m > n$ then $Q_m \parallel Q_n = Q_n$. Gathering all the terms in Q_n and using the fact that the probabilities of identical choices add up we get

$$\prod_{q_n} \prod_{p_k} (Q_{n+1} \parallel Q_k) \;\equiv\; \prod_{r_n} Q_n \quad \text{where}$$

$$r_0 = p_0,\ r_{n+1} = q_n(\sum_{k=n+1}^{\infty} p_k) + (\sum_{k=n+1}^{\infty} q_k) p_{n+1}.$$

Substituting for q_n and for $\sum_{k=n+1}^{\infty} q_k = 1 - \sum_{k=0}^{n} q_k = \prod_{j=0}^{n}(1 - \sum_{k=0}^{j} p_k)$ we get

$$\begin{aligned} r_{n+1} &= (\prod_{j=0}^{n-1}(1 - \sum_{k=0}^{j} p_k)\sum_{k=0}^{n} p_k)(\sum_{k=n+1}^{\infty} p_k) + \prod_{j=0}^{n}(1 - \sum_{k=0}^{j} p_k)\ p_{n+1} \\ &= \prod_{j=0}^{n}(1 - \sum_{k=0}^{j} p_k)\sum_{k=0}^{n+1} p_k \ = \ q_{n+1}. \end{aligned}$$

So the antecedent is true. We deduce that

$$P \ \equiv \ \mu X \bullet ((a \rightarrow X) \parallel Q) \ \equiv \ \prod_{q_n} Q_n.$$

At first it may seem surprising that the probabilities with which P chooses a certain number of $a's$ are different from the probabilities with which Q chooses bnt this is explained by the fact that P chooses not only once, but several times over. For instance, supposing that to begin with P chooses to do three a's. After it has done one a it chooses again and runs this second choice in parallel with the first. If the second choice is of less than two a's, P can't do the three a's which it originally set out to do.

5.6 Probabilistic vs. Non-deterministic Choice

In the two examples in this section we take a process in tbe failures-divergences model and compare it with its probabilistic analogue, whicb we obtain by snbstituting probabilistic choice for non-deterministic choice. We have not attempted to formalise the relation between the probabilistic and the failures-divergences model, but these examples show some important differences between the two models which make it unlikely that the models can be related to each other by a simple abstraction function.

The first example shows the difference between non-deterministic and probabilistic choice as far as asymptotic behaviour is concerned. Consider the process $P = 0 \rightarrow P \sqcap 1 \rightarrow P$ in the failures- divergences model. This can choose to perform 1 forever; so hiding 1 would lead to divergence. By contrast, the $PCSP_0$-process $P = 0 \rightarrow P \ {}_p\sqcap\ 1 \rightarrow P$ where $p \neq 0$ is fair in the sense that almost all traces of P contain 0 infinitely often (as shown in lemma 5.1.1). So we would expect that hiding 1 would result in the process which performs 0 forever, $Q = 0 \rightarrow Q$, say. This is indeed the case: consider the probability of the set of sequences starting with n 0's after hiding 1.

$$\begin{aligned} [\![P \backslash \{1\}]\!] S\langle 0\rangle^n &= [\![P]\!](\bigcup_{i_1=0}^{\infty}\bigcup_{i_2=0}^{\infty}\cdots\bigcup_{i_n=0}^{\infty} S(\langle 1\rangle^{i_1}\langle 0\rangle\langle 1\rangle^{i_2}\langle 0\rangle\ldots\langle 1\rangle^{i_n}\langle 0\rangle)) \\ &= \sum_{i_1}(1-p)^{i_1}p\ \sum_{i_2}(1-p)^{i_2}p\ \ldots\sum_{i_n}(1-p)^{i_n}p \\ &= 1. \end{aligned}$$

From the fact that $S\langle 0\rangle^* \downarrow \{\langle 0\rangle^\omega\} \Rightarrow [\![P\backslash\{1\}]\!]\, S\langle 0\rangle^* \downarrow [\![P\backslash\{1\}]\!]\, \{\langle 0\rangle^\omega\}$ it follows that $[\![P\backslash\{1\}]\!]\, \{\langle 0\rangle^\omega\} = 1$. Therefore all sets not containing the infinite sequence of 0's have probability zero and $\forall A \in \Omega$

$$\begin{aligned} [\![P\backslash\{1\}]\!]\, A &= \begin{cases} 1 & \text{if } \langle 0\rangle^\omega \in A \\ 0 & \text{otherwise} \end{cases} \\ &= [\![Q]\!]\, A. \end{aligned}$$

So $P\backslash\{1\} \equiv Q$.

Our second example shows that using probabilistic choice eliminates the problems of unbounded non-determinism. The problems caused by unbounded non-determinism in the failures-divergences model are demonstrated by the following process. Let

$$Q_0 = STOP,\ Q_{n+1} = a \rightarrow Q_n \text{ and } P_n = \bigsqcap_{i>n} Q_i.$$

Then $P_n \sqsubseteq P_{n+1}$, $\bigsqcup_{n=1}^{\infty} P_n = STOP$ and $P_n\backslash\{a\} = CHAOS$ so that $\bigsqcup(P_n\backslash\{a\}) \neq (\bigsqcup P_n)\backslash\{a\}$. In the probabilistic analogue the problem disappears. Let

$$P_n \mathrel{\hat{=}} \bigsqcap_{p_{n,i}\cdot n} Q_i \quad \text{where } \forall k \cdot \textstyle\sum_n p_{n,k} = 1.$$

Now $\lim [\![P_n]\!] = [\![\, \mu X \bullet a \rightarrow X \,]\!]$ so that

$$\lim [\![P_n\backslash\{a\}]\!] \;=\; \lim [\![STOP]\!] \;=\; [\![STOP]\!] \;=\; [\![(\mu X \bullet a \rightarrow X)\backslash\{a\}]\!].$$

5.7 Discussion

We have defined the semantics of a probabilistic language which features a subset of the operators of *CSP*, with probabilistic choice substituted for non-deterministic choice. Processes are defined as probability measures on infinite sequences of actions and operators as transformations or linear combinations of measures.

We have given examples which show that this semantics enables us to reason about important properties such as liveness, asymptotic frequencies of actions, fairness and probabilistic correctness. We have also proved the validity of algebraic laws which are important for reasoning about parallel systems in general. The laws of $PCSP_0$ are the same as those of the corresponding subset of operators of the traces model of *CSP*, with two exceptions: parallel composition is idempotent in the traces model, but not in $PCSP_0$, and unguarded recursion may he well-defined in $PCSP_0$, but not in the traces model. Like the traces model, the probabilistic model does uot distinguish between deadlock and divergence; the infinite sequence of unobservable events is used to model both.

It would be interesting to determine the precise relation between $PCSP_0$ and other models of *CSP*. It is possible that the relation between $PCSP_0$ and the traces model could be characterised by a projection function which maps each set of extensions of a finite trace of positive probability to a trace of the corresponding process in the traces model, but we have not looked at this in detail.

The main disadvantage of $PCSP_0$, which limits its usefulness, is that it lacks the operators for general parallel composition and external choice. This problem is addressed in the remainder of the thesis.

Chapter 6

Alphabetised Parallel Composition

In this chapter we investigate a way of defining the parallel composition of processes which syuchrouise on only some actions and perform others independently. The relative order of unsyuchronised actions is arbitrary, which means that parallel composition in general cannot be characterised by a function which maps pairs of traces into single ones, but only by a relation between pairs of traces and possible interleavings of unsynchronised actions.

Recall that given a measure P and a *function* F a new measure P' can be defined by setting

$$P'A \;\widehat{=}\; P\,F^{-1}A$$

for all $A \in \mathcal{F}$. For a function the inverse images of disjoint sets are disjoint or, equivalently,

$$\forall A \in \mathcal{F} \;\cdot\; F^{-1}A \cap F^{-1}A^c = \emptyset$$

so that for any disjoint sets A and B

$$\begin{aligned} P'(A \cup B) &= P\,F^{-1}(A \cup B) = P\,(F^{-1}A \cup F^{-1}B) = P\,F^{-1}A + P\,F^{-1}B \\ &= P'A + P'B. \end{aligned}$$

If F is a relation disjoiutedness is not always preserved under the inverse image which, at first sight, means that it cannot be nsed for a transformation of measure. However, we will show that the suhset of sets for which disjointedness is preserved forms a σ-field, $\mathcal{F}'$ say. Therefore we can use the definition above (with F a relation) to define a probability measure P' on the restricted σ-field $\mathcal{F}'$.

This and other results about the transformation of measure with relations form the first section of this cbapter. We will use these in the second section, where we

define the semantics of an extended version of $PCSP_0$ which includes alphabetised parallel composition, based on a transformation relation. In the third section we give an example to show how to use the extended model. The last section we discuss the advantages and disadvantages of this approach.

6.1 Transformation of Measure with Relations

In this section we will prove that the sets whose disjointedness is preserved under the inverse image of a relation form a σ-field. In the first lemma these sets are described as the sets whose inverse image is disjoint from the inverse image of their complement. In the second and third lemma we find alternative representations of this σ-field which enable us to identify sets in this σ-field more easily.

Lemma 6.1.1 Given measurable spaces $(\Omega, \mathcal{F})$ and $(\Omega', \mathcal{F}')$, and a relation $R : \Omega \leftrightarrow \Omega'$ the collection of sets

$$\mathcal{M} \;\hat{=}\; \{A : \mathcal{F}' \mid R^{-1}A \cap R^{-1}A^c = \emptyset \wedge R^{-1}A \in \mathcal{F}\}$$

is a σ-field. □

Proof The class $\mathcal{M}$ is obviously closed under complementation. We show that $\mathcal{M}$ is also closed under finite unions and that it is a monotone class. Let $A, B \in \mathcal{M}$. Then

$$\begin{aligned}
&R^{-1}(A \cup B) \cap R^{-1}(A \cup B)^c \\
&= (R^{-1}A \cup R^{-1}B) \cap R^{-1}(A \cup B)^c \\
&= (R^{-1}A \cap R^{-1}(A \cup B)^c) \cup (R^{-1}B) \cap R^{-1}(A \cup B)^c) \\
&\subseteq (R^{-1}A \cap R^{-1}A^c) \cup (R^{-1}B \cap R^{-1}B^c) \\
&= \emptyset.
\end{aligned}$$

and $R^{-1}(A \cup B) = R^{-1}A \cup R^{-1}B \in \mathcal{F}$. So $A \cup B \in \mathcal{M}$. Let $(B_i)_{i:\mathbb{N}}$ be a sequence of sets in $\mathcal{M}$ such that $B_n \uparrow B$. Then

$$\begin{aligned}
R^{-1}B \cap R^{-1}B^c &= R^{-1}\textstyle\bigcup_n B_n \cap R^{-1}\bigcap_n B_n^c \\
&= \textstyle\bigcup_n R^{-1}B_n \cap R^{-1}\bigcap_n B_n^c \\
&= \textstyle\bigcup_n (R^{-1}B_n \cap R^{-1}\bigcap_m B_m^c) \\
&\subseteq \textstyle\bigcup_n (R^{-1}B_n \cap R^{-1}B_n^c) \\
&= \emptyset.
\end{aligned}$$

Also $R^{-1}B = \bigcup_n R^{-1}B_n \in \mathcal{F}$. So $B \in \mathcal{M}$, $\mathcal{M}$ is a monotone class and hence a σ-field. □

Note that $\mathcal{M}$ is always non-empty, but that it can be trivial.

Lemma 6.1.2 Let $\mathcal{M}_1 \triangleq \{A : \mathcal{F}' \mid (R^{-1}; R)(A \cap \text{ran } R) = (A \cap \text{ran } R) \in \mathcal{F}\}$. Then $\mathcal{M}_1 = \mathcal{M}$. □

Proof By the definition of inverse $x\ R\ y \Leftrightarrow y\ R^{-1} x$ which implies

$$B \cap R\ A \neq \emptyset \quad \Leftrightarrow \quad R^{-1} B \cap A \neq \emptyset. \tag{6.1}$$

We first show that $\mathcal{M}_1 \subseteq \mathcal{M}$. For $A \in \mathcal{M}_1$

$(R^{-1}; R)(A \cap \text{ran } R) = (A \cap \text{ran } R)$
$\Rightarrow$ $(R^{-1}; R)(A \cap \text{ran } R) \cap (A^c \cap \text{ran } R) = \emptyset$
because $(A \cap \text{ran } R) \cap (A^c \cap \text{ran } R) = \emptyset$
$\Rightarrow$ $(A \cap \text{ran } R) \cap (R^{-1}; R)(A^c \cap \text{ran } R) = \emptyset$
by 6.1
$\Rightarrow$ $(R^{-1}; R)(A \cap \text{ran } R) \cap (R^{-1}; R)(A^c \cap \text{ran } R) = \emptyset$
by assumption
$\Rightarrow$ $R^{-1}(A \cap \text{ran } R) \cap R^{-1}(A^c \cap \text{ran } R) = \emptyset$
$\Rightarrow$ $R^{-1} A \cap R^{-1} A^c = \emptyset$.

Conversely, to show that $\mathcal{M} \subseteq \mathcal{M}_1$ we show that $A \notin \mathcal{M}_1 \Rightarrow A \notin \mathcal{M}$.

$A \notin \mathcal{M}_1$
$\Leftrightarrow$ $(R^{-1}; R)(A \cap \text{ran } R) \supset A \cap \text{ran } R$
$\Rightarrow$ $(R^{-1}; R)(A \cap \text{ran } R) \cap (A \cap \text{ran } R)^c \neq \emptyset$
$\Rightarrow$ $R^{-1}(A \cap \text{ran } R) \cap R^{-1}(A \cap \text{ran } R)^c \neq \emptyset$
$\Rightarrow$ $R^{-1} A \cap R^{-1}(A^c \cup (\text{ran } R)^c) \neq \emptyset$
$\Rightarrow$ $R^{-1} A \cap R^{-1} A^c \neq \emptyset$
since $R^{-1} A \supseteq R^{-1}(A \cap \text{ran } R)$ and $R^{-1}(A^c \cup (\text{ran } R)^c) = R^{-1} A^c$.

So equality holds. □

Yet another equivalent definition of $\mathcal{M}$ is formulated in terms of the transitive closure of $(R^{-1}; R)$. For $w \in \Omega$ let

$$T\ w \quad \triangleq \quad \bigcup_{n \geq 0} (R^{-1}; R)^n w.$$

Lemma 6.1.3 Let $\mathcal{M}_2 = \{T\ A \mid A \in \mathcal{F}' \wedge R^{-1} A \in \mathcal{F}\}$. Then $\mathcal{M}_2 = \mathcal{M}_1$. □

Proof

$$\begin{aligned} A \in \mathcal{M}_1 &\Leftrightarrow (R^{-1}\,;\, R)(A \cap \operatorname{ran} R) = (A \cap \operatorname{ran} R) \\ &\Leftrightarrow T\,A = A \\ &\Leftrightarrow A \in \mathcal{M}_2. \end{aligned}$$

□

6.2 The Extended Model

If we are to add an operator based on a transformation relation to our model then the σ-field on which a process is defined must be made explicit. In the following we define a σ-field for every existing $PCSP_0$ construct, writing $\sigma[\![P]\!]$ for the σ-field on which the measure denoted by P is defined. Apart from being parametrised by a σ-field the definitions of processes and operators remain unchanged. So to prove that all the $PCSP_0$-laws still hold in the extended model we only need to add proofs concerning the equality of the σ-fields. Where the law depends on the commutativity of two transformation functions, f and g say, equality of the σ-fields follows immediately. Therefore we need to reconsider only the proofs of those laws which do not follow from the commutativity of transformation functions.

Having determined the effect of relational transformations on $PCSP_0$ we define alphabetised parallel composition based on a relation, $merge_{B,C}$, and show that it satisfies all the laws we would expect it to hold.

The Semantics of $PCSP_0$ with Variable σ-fields

The processes $STOP$ and $SKIP$ are defined on the standard σ-field.

$$\begin{aligned} \sigma[\![STOP]\!] &\;\widehat{=}\; \mathcal{F} \\ \sigma[\![SKIP]\!] &\;\widehat{=}\; \mathcal{F}. \end{aligned}$$

The σ-field of $a \rightarrow P$ contains all the standard sets of traces not beginning with a and the sets of $\sigma[\![P]\!]$ prefixed by a.

$$\sigma[\![a \rightarrow P]\!] \;\widehat{=}\; \{A : \mathcal{F} \mid prefix_a^{-1} A \in \sigma[\![P]\!]\}.$$

$P \;{}_p\!\sqcap\; Q$ is defined only for arguments with identical σ-fields: let $\sigma[\![P]\!] = \sigma[\![Q]\!]$. Then

$$\sigma[\![P \;{}_p\!\sqcap\; Q]\!] \;\widehat{=}\; \sigma[\![P]\!] (= \sigma[\![Q]\!])$$

For any law involving probabilistic choice, equality of the σ-fields on the right and left-hand sides of the equation is obvious. The reason we do not allow probabilistic

choice between processes with differing σ-fields, is that even though it could be defined on the intersection of these fields, this would distroy law 3, namely that $P \sqcap Q = P$.

The σ-field of $X \backslash B$, where $B \subseteq \Sigma$, must be such that unhiding B yields a set in $\sigma[\![P]\!]$.

$$\sigma[\![P \backslash B]\!] \;\widehat{=}\; \{A : \mathcal{F} \mid hide_B^{-1} \in \sigma[\![P]\!]\}.$$

To prove that $\sigma[\![P \backslash \Sigma]\!] = \sigma[\![STOP]\!]$ note that since $hide_\Sigma^{-1} A = \Omega$ if $\langle\tau\rangle^\omega \in A$, $\emptyset$ otherwise, and Ω and $\emptyset$ are contained in any σ-field on Ω it follows that $\{A : \mathcal{F} hide_\Sigma^{-1} \in \sigma[\![P]\!]\} = \mathcal{F}$. We have $\sigma[\![P \backslash \emptyset]\!] = \sigma[\![P]\!]$ because $hide_\emptyset = id$.

The σ-field for simple parallel composition is

$$\sigma[\![P \| Q]\!] \;\widehat{=}\; \{A : \mathcal{F} \mid par^{-1} A \in \sigma[\![P]\!] \times \sigma[\![Q]\!]\}$$

It will turn out that simple parallel composition is a special case of alphabetised parallel composition. So we need not prove its laws separately.

The sequential composition of P and Q has σ-field

$$\sigma[\![P \,;\, Q]\!] \;\widehat{=}\; \{A : \mathcal{F} \mid seq^{-1} A \in \sigma[\![P]\!] \times \sigma[\![Q]\!]\}$$

To show that $\sigma[\![SKIP \,;\, P]\!] = \sigma[\![P]\!]$ recall that $SKIP \backslash (\Sigma - \{\checkmark\}) = SKIP$. This allows us to write

$$\begin{aligned}
& A \in \sigma[\![SKIP \,;\, P]\!] \\
& \quad \Leftrightarrow \; (seq^{-1} \,;\, (hide_{\Sigma-\{\checkmark\}}^{-1}, id)) A \in \mathcal{F} \times \sigma[\![P]\!] \\
& \quad \Leftrightarrow \; (hide_{\Sigma-\{\checkmark\}}^{-1} \,;\, id)((seq^{-1} A) \cap \mathrm{ran}(hide_{\Sigma-\{\checkmark\}}, id)) \in \mathcal{F} \times \sigma[\![P]\!] \\
& \quad \Leftrightarrow \; (hide_{\Sigma-\{\checkmark\}}^{-1} \,;\, id)(\{\langle\checkmark\rangle\langle\tau\rangle^\omega\} \times A \;\cup\; \{\langle\tau\rangle^\omega\} \times \Omega) \in \mathcal{F} \times \sigma[\![P]\!] \\
& \qquad \text{assuming w.o.l.o.g. that } \langle\tau\rangle^\omega \in A \\
& \quad \Leftrightarrow \; A \in \sigma[\![P]\!].
\end{aligned}$$

So $\sigma[\![SKIP \,;\, P]\!] = \sigma[\![P]\!]$. A similar argument can be used to show that $\sigma[\![P \,;\, SKIP]\!] = \sigma[\![P]\!]$ and that $\sigma[\![STOP \,;\, P]\!] = \sigma[\![STOP]\!]$.

The σ-field for interleaving is

$$\sigma[\![P \;|||\; Q]\!] \;\widehat{=}\; \{A : \mathcal{F} \mid interleave^{-1} A \in \sigma[\![P]\!] \times \sigma[\![Q]\!] \times \mathcal{F}\}$$

The only law for which we need to prove equality of the σ-fields is $STOP_p \;|||\; P \equiv P$. This proof is very similar to the proof that $SKIP \,;\, P \equiv P$.

If we allowed relational transformations in recursive process definitions then each unfolding of the recursion could change the underlying σ-field. However, convergence is defined only for sequences of measures which have the same underlying σ-field. So we allow recursive processes only if they are defined on the full σ-field $\mathcal{F}$. That is, we cannot have recursive calls to parallel processes unless they are fully synchronised or form a master-slave system (where the actions of one component are a subset of the actions of the other).

Alphabetised Parallel Composition

Alphabetised parallel composition is denoted by

$$P \;{}_B\|_C\; Q$$

where B, C are sets of actions. Both contain τ. The processes $P$and Q synchronise only on actions in the intersection of B and C. Events outside B happen without the participation of P, with a probability and ordering entirely determined by Q, and events outside C happen without the participation of Q, with a probability entirely determined by P. The semantics of this operator is defined as a transformation of the product measure $([\![P]\!]\rho \times [\![Q]\!]\rho)$ with the relation $merge_{B,C}$:

$$\begin{aligned}
&\sigma[\![P \;{}_B\|_C\; Q]\!] \\
&\quad \widehat{=}\ \{A : \mathcal{F} \mid merge_{B,C}^{-1} A \cap merge_{B,C}^{-1} A^c = \emptyset \wedge merge_{B,C}^{-1} \in \sigma[\![P]\!] \times \sigma[\![Q]\!]\} \\
&\forall A \in \sigma[\![P \;{}_B\|_C\; Q]\!] \cdot [\![P \;{}_B\|_C\; Q]\!]\rho\ A \ \widehat{=}\ ([\![P]\!]\rho \times [\![Q]\!]\rho)\ merge_{B,C}^{-1} A.
\end{aligned}$$

The relation $merge_{B,C}$ maps a pair of traces to the longest trace up to which they agree on the order of actions in $B \cap C$.

$$\begin{aligned}
&merge_{B,C} : \Omega \times \Omega \leftrightarrow \Omega \\
&\forall u, v \in \Omega \cdot \\
&\quad w \in merge_{B,C}(u, v) \\
&\quad \Leftrightarrow\ (w \in (B \cup C)^\omega \wedge w \lceil B \leq u \wedge w \lceil C \leq v) \\
&\qquad \vee (w = t\langle\tau\rangle^\omega \wedge t \in (B \cup C)^* \wedge t \lceil B < u \wedge t \lceil C < v \\
&\qquad\quad \wedge \forall e \in (B \cup C) \cdot (t\langle e\rangle) \lceil B \not\leq u \vee (t\langle e\rangle) \lceil C \not\leq v).
\end{aligned}$$

If w and w' are two possible mergings of the traces u, v then $merge_{B,C}^{-1}\{w\} \cap merge_{B,C}^{-1}\{w'\}$ is not empty. So the restriction of the σ-field on which parallel composition is defined means that in a parallel system probabilities are known only for the set of all possible orderings of unsynchronised actions, but not for individual orderings within that set.

Note that $merge_{\Sigma_\tau,\Sigma_\tau} = par$ and that if $B \subseteq C$ or $C \subseteq B$ then $merge_{B,C}$ is a function. We could have used a slightly different definition in which a merging w of two traces u, v has to satisfy $w \lceil B = u \wedge w \lceil C = v$. The difference matters only if u and v have tails of unsyuchronised actions in $B - C$ and $C - B$ respectively. The definition we use allows mergings in which actions in u always have precedence over actions in v (or the other way round). Thus a sequence w containing only a finite number of actions in v and the same tail as u would be a possible merging. The alternative definition excludes this possibility. It implies that infiuite overtaking would always have zero probability simply because the traces resulting from infinite

overtaking would be outside the range of *merge*. We have opted for a relation which does not have this implicit fairness property.

The laws for alphabetised parallel composition are the same as those in the traces model of *CSP* with one important exception: associativity holds only in special cases.

Lemma 6.2.1

L1 $P \,{}_B\|_C\, Q \equiv Q \,{}_B\|_C\, P.$

L2 $B \subseteq C \Rightarrow P \,{}_B\|_C\, STOP \equiv STOP.$

L3 $B \subseteq C \subseteq D \Rightarrow$
$P \,{}_B\|_{C \cup D}\, (Q \,{}_C\|_D\, O) \equiv (P \,{}_B\|_C\, Q) \,{}_{B \cup C}\|_D\, O.$

L4 $(P \,{}_p\sqcap\, Q) \,{}_B\|_C\, O \equiv P \,{}_B\|_C\, O \,{}_p\sqcap\, Q \,{}_B\|_C\, O.$

L5 $P \,{}_B\|_C\, (Q \,{}_p\sqcap\, O) \equiv P \,{}_B\|_C\, Q \,{}_p\sqcap\, P \,{}_B\|_C\, O.$

L6 $a \in B \cap C \Rightarrow$
$(a \rightarrow P) \,{}_B\|_C\, (a \rightarrow Q) \equiv a \rightarrow (P \,{}_B\|_C\, Q).$

L7 $a \in B \cap C, b \in B - C \Rightarrow$
$(b \rightarrow P) \,{}_B\|_C\, (a \rightarrow Q) \equiv b \rightarrow (P \,{}_B\|_C\, a \rightarrow Q).$

L8 $a, b \in B \cap C, a \neq b \Rightarrow$
$(b \rightarrow P) \,{}_B\|_C\, (a \rightarrow Q) \equiv STOP.$

□

Proof Law 1 follows from the symmetry of *merge* and Fubini's theorem. For law 2 note that since $\operatorname{ran} hide_\Sigma = \{\langle\tau\rangle^\omega\}$ and $merge_{B,C}(u, \langle\tau\rangle^\omega) = \langle\tau\rangle^\omega$ for all $u \in \Omega$ it follows that $(id, hide_\Sigma)\,;\, merge_{B,C}$ is a function. Thus

$$\sigma[\![P \,{}_B\|_C\, STOP]\!] = \sigma[\![P \,{}_B\|_C\, (STOP\backslash_\Sigma)]\!] = \mathcal{F} = \sigma[\![STOP]\!].$$

For all sets $A \in \mathcal{F}$ we have

$$\begin{aligned}
[\![P \,{}_B\|_C\, STOP]\!]A &= ([\![P]\!] \times [\![STOP]\!])\; merge^{-1}_{B,C}\, A \\
&= ([\![P]\!] \times [\![STOP]\!])\; (merge^{-1}_{B,C}\, A \cap (\Omega \times \{\langle\tau\rangle^\omega\})) \\
&\quad \text{since } ([\![P]\!] \times [\![STOP]\!])(\Omega \times \{\langle\tau\rangle^\omega\}) = 1 \\
&= \begin{cases} [\![P]\!]\,\Omega\,[\![STOP]\!]\,\{\langle\tau\rangle^\omega\} & \text{if } \langle\tau\rangle^\omega \in A \\ 0 & \text{otherwise} \end{cases} \\
&\quad \text{since } \langle\tau\rangle^\omega \in A \Leftrightarrow merge^{-1} A \cap (\Omega \times \{\langle\tau\rangle^\omega\}) \neq \emptyset \\
&= [\![STOP]\!]\, A.
\end{aligned}$$

To see why parallel composition with *merge* is not always associative consider the composition of three processes on sets B, C and D where $(B \cup C) \subseteq D$. Then $(merge_{B,C}, id); merge_{B\cup C,D}$ is a relation whereas $(id, merge_{C,D}); merge_{B,D}$ is a function. Hence the two sides cannot be the same. Only if $B \subseteq C \subseteq D$ are the transformations on both sides functions. In this special case we can write for all $u, v, w \in \Omega$

$$
\begin{aligned}
& merge_{B\cup C,D}(merge_{B,C}(u,v), w) \\
&= \begin{cases} merge_{C,D}(v,w) & \text{if } v \in C^\omega \wedge v\restriction B = u \\ merge_{C,D}(t\langle\tau\rangle^\omega, w) & \text{if } t \in C^* \wedge t\restriction B < u \wedge t < v \\ & \qquad \wedge \forall e \in C \cdot t\langle e\rangle\restriction B \not< u \vee t\langle e\rangle \not< v \end{cases} \\
&= \begin{cases} w & \text{if } w \in D^\omega \wedge w\restriction C \leq v \wedge w\restriction B \leq u \\ t\langle\tau\rangle^\omega & \text{if } t \in D^* \wedge t\restriction C < v \wedge t\restriction B < u \\ & \qquad \wedge \forall e \in D \cdot t\langle e\rangle\restriction B \not< u \vee t\langle e\rangle\restriction C \not< v \vee t\langle e\rangle \not< v. \end{cases}
\end{aligned}
$$

Evaluation of $merge_{B,C\cup D}(u, merge_{C,D}(v,w))$ leads to the same expression.

Probabilistic choice is defined only between processes with identical σ-fields. So for law 4 assume that $\sigma[\![P]\!] = \sigma[\![Q]\!]$. Then

$$
\begin{aligned}
& \sigma[\![(P \,{}_p\sqcap\, Q) \,{}_B\|_C\, O]\!] \\
&= \{A : \mathcal{F} \mid merge_{B,C}^{-1} A \cap merge_{B,C}^{-1} A^c = \emptyset \wedge merge_{B,C}^{-1} \in \sigma[\![P]\!] \times \sigma[\![O]\!]\} \\
&= \sigma[\![(P \,{}_B\|_C\, O) \,{}_p\sqcap\, (Q \,{}_B\|_C\, O)]\!].
\end{aligned}
$$

The proof of equality in probability follows along the same lines as the proof in section 3.5 that probabilistic choice distributes over simple parallel composition. Similarly for law 5.

Law 6 follows if we can show that for $a \in B \cap C$

$$(prefix_a, prefix_a); merge_{B,C} \;=\; merge_{B,C}; prefix_a.$$

This is true because $\forall\, u, v \in \Omega$

$$
\begin{aligned}
& w \in (merge_{B,C}(prefix_a, prefix_a))(u,v) \\
& \Leftrightarrow \; w \in merge_{B,C}(\langle a\rangle u, \langle a\rangle v) \\
& \Leftrightarrow \; (w \in (B\cup C)^\omega \wedge w\restriction B \leq \langle a\rangle u \wedge w\restriction C \leq \langle a\rangle v) \\
& \quad \vee (w = t\langle\tau\rangle^\omega \wedge t \in (B\cup C)^* \wedge t\restriction B < \langle a\rangle u \wedge t\restriction C < \langle a\rangle v \\
& \quad \wedge \forall e \in (B\cup C).(t\langle e\rangle)\restriction B \not< \langle a\rangle u \vee (t\langle e\rangle)\restriction C \not< \langle a\rangle v) \\
& \Leftrightarrow \; w = \langle a\rangle w' \wedge w' \in merge_{B,C}(u,v) \\
& \Leftrightarrow \; w \in prefix_a(merge_{B,C}(u,v)).
\end{aligned}
$$

Similarly, it can be shown that if $a \in B \cap C$ and $b \in B - C$ then

$$(prefix_b, prefix_a); merge_{B,C} \;=\; (id, prefix_a); merge_{B,C}; prefix_b$$

and law 7 follows. For law 8 note that if $a \neq b$ and $a, b \in B \cap C$ then for all $u, v \in \Omega$

$$(merge_{B,C}(prefix_a, prefix_b))(u, v) = \langle\tau\rangle^\omega.$$

So

$$[\![(b \to P) \,{}_B\|_C\, (a \to Q)]\!]A = \begin{cases} ([\![P]\!] \times [\![Q]\!])\ \Omega \times \Omega & \text{if } \langle\tau\rangle^\omega \in A \\ ([\![P]\!] \times [\![Q]\!])\ \emptyset & \text{otherwise} \end{cases}$$
$$= [\![STOP]\!]A.$$

□

6.3 The Loss Rate of a Pipe

Consider a pipe of two media with loss rates p and q. (The loss rate is the long term or asymptotic frequency with which a medium loses data.) We model the media in the same way as in the example of section 5.1.

$$P = in \to (P \,{}_p\sqcap\, mid \to P)$$
$$Q = mid \to (Q \,{}_q\sqcap\, out \to Q).$$

In a way similar to the example of section 5.2 it can be shown that these two processes have indeed the required loss rates. The process $P \,{}_B\|_C\, Q$ where $B = \{in, mid\}$, $C = \{mid, out\}$, forms a pipe which inputs data on channel in, passes it on internally on channel mid and outputs it on channel out. We would like to know the overall loss rate of the pipe.

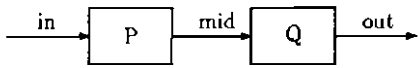

Figure 6.1: The pipe

The loss rate equals 1 minus the success rate, which is the asymptotic ratio of outputs to inputs in the infinite traces. To determine the latter consider the set $R(i, j)$ of traces which contain j internal communications before the first output and i inputs before the jth internal communication. Any such trace has a prefix t such that

$$t_{\#t-1} = out \ \wedge\ t{\downarrow}\{out\} = 1 \ \wedge\ t{\downarrow}\{mid\} = j \ \wedge$$
$$t_{i+j-1} = mid \ \wedge\ (t/i+j){\downarrow}\{mid\} = 0 \ \wedge\ (t{\uparrow}i+j){\downarrow}\{in\} = i.$$

Then the smallest superset of $S(t)$ to which we can assign a probability is

$$\begin{aligned} T\,S(t) \;=\; \{s\,u \mid (s\restriction B = t\restriction B \wedge s\restriction C = t\restriction C) \\ \vee\, (s\restriction B = (t\restriction i+j)\restriction B \wedge s\restriction C = (t\restriction i+j)\restriction C \wedge u\downarrow\{mid\} = 0)\}. \end{aligned}$$

Then $R(i,j) = \bigcup_t T\,S(t)$ (where t as above) is the smallest measurable set containing all the traces we are interested in. Note for example that $R(2,1)$ contains traces beginning with $\langle in, in, mid, in, out\rangle$ where the last *in* has nothing to do with the output but just happened to be input while the previous message was still in the pipe. It also contains $\langle in, mid\rangle\langle in\rangle^\omega$.

$$\begin{aligned} [\![P\ {}_B\|_C\ Q]\!]\ R(i,j) &= ([\![P]\!]\times[\![Q]\!])\ merge^{-1}_{B,C} R(i,j) \\ &= ([\![P]\!]\times[\![Q]\!])\ (A_1\times S(\langle mid\rangle^j\langle out\rangle)\ \cup\ A_2\times S(\langle mid\rangle^j)) \end{aligned}$$

where

$$\begin{aligned} A_1 &= \{s\,u \mid \#s = i+j \wedge s_{i+j-1} = mid \wedge s\downarrow\{mid\} = j \wedge s\downarrow\{in\} = i\} \\ A_2 &= \{s\langle in\rangle^\omega \mid \#s = i+j \wedge s_{i+j-1} = mid \wedge s\downarrow\{mid\} = j \wedge s\downarrow\{in\} = i\}. \end{aligned}$$

Since $[\![P]\!]A_2 = 0$ we are left with

$$[\![P]\!]A_1\ [\![Q]\!]\ S(\langle mid\rangle^j\langle out\rangle) \;=\; \binom{j+(i-j)-1}{i-j} p^{i-j}(1-p)^j\ q^{j-1}(1-q).$$

Taking the sum over all j then gives the probability of i inputs being needed to produce an output. Let U be a random variable recording the number of *in*'s up to the first *out*. Then U has expectation

$$\begin{aligned} E(U) &= \sum_{i=1}^{\infty}\sum_{j=1}^{i}\binom{j+(i-j)-1}{i-j} p^{i-j}(1-p)^j\ q^{j-1}(1-q)\ i \\ &= \sum_{j=1}^{\infty}\sum_{i=0}^{\infty}\binom{j+i-1}{i} p^i(1-p)^j\ q^{j-1}(1-q)\ (i+j) \\ &= \sum_{j=1}^{\infty}(1-p)^j\ q^{j-1}(1-q)\ j\sum_{i=0}^{\infty}\binom{j+i}{i} p^i \\ &= \frac{(1-q)}{(1-p)}\sum_{j=0}^{\infty} q^j\ (j+1) \\ &= \frac{1}{(1-q)(1-p)}. \end{aligned}$$

Thus the success rate of the pipe is $1/E(U) = (1-q)(1-p)$ and the loss rate is $p+q-pq$ (which is as we would have expected from combinatorial arguments).

6.4 Discussion

We have presented an extension of $PCSP_0$ which includes an operator for alphabetised parallel composition. This operator is defined as a transformation of measure based on the relation *merge*. We have shown that a relation can be used for a transformation of measure if restricted those sets for which probabilities are defined. For a system of parallel processes this means that we can no longer assign a probability to every set in $\mathcal{F}$, but only to sets which contain all alternative interleavings of unsynchronised actions. We have given an example to show that this still allows us to deduce an interesting property of a parallel system from the properties of its components.

However, even though the extended model enables us to reason about a wider class of processes than $PCSP_0$ it is not really useful without an operator for external choice. If a process offers external choice we can say that with probability 1 it will do one of two actions, but we cannot a priori give the probability of either one being chosen. We could assign probabilities to those sets only which contain both branches of the choice, but for most processes this will leave us with little more than the trivial σ-field $\{\Omega, \emptyset\}$. So in the next chapter we will try a different approach, which allow us to define both alphabetised parallel composition and external choice.

Chapter 7

A Model with External Choice

In *CSP* the term $e : E \to P_e$ denotes a process which offers *deterministic* or *external* choice. Such a process will participate in whatever action e its environment offers, as long as e is in E, and then behave like P_e. If the environment offers an action outside E the system will deadlock. This behaviour cannot be described in $PCSP_0$, because the probability with which a $PCSP_0$-process decides what to do is always independent of its environment. External choice requires a notion of dependence or conditioning on the environment. In this chapter we formalise this notion by defining a process as a conditional probability measure. The idea is that if a process P is offered a sequence $y \in \Omega$ by its environment, we know the probability with which it performs a set $A \in \mathcal{F}$. We use this to define the semantics of a second language, $PCSP$, which differs from $PCSP_0$ in that it contains operators for external choice and alphabetised parallel composition, but lacks the operators for sequential composition, hiding and interleaving.

The syntax of the language $PCSP$ contains the following constructs:

$$P ::= STOP \mid X \mid a \to P \mid P \sqcap_p Q \mid e : E \to P_e \mid P \Box_s Q \mid$$
$$P \parallel Q \mid P \parallel_B^C Q \mid f(P) \mid \mu X \bullet P \mid \langle X_i = P_i \rangle$$

Let CM be the space of conditional probability measures. Like the semantics of other models the semantics of a $PCSP$-term P is parametrised by a binding for its free variables. Let BND be the domain of all bindings of variables to cpm's:

$$BND \triangleq VAR \to CM$$

We use round brackets $(\!|\ |\!)$ for a semantic function which defines the meaning of $PCSP$ terms:

$$(\!|\ |\!) \triangleq PCSP \to BND \to CM$$

We will show that the cpm's representing processes have two additional properties:

1. If offered y by its environment, a process P can either perform y or deadlock at some point. So

$$P(\{y\} \cup \bigcup_n \{(y\lceil n)\langle\tau\rangle^\omega\}, y) \;=\; 1.$$

2. For fixed $t \in \Sigma_\tau^*$ the probability $P\ (S(t), y)$ is constant if y ranges over the set of extensions of t. Intuitively, the probability of n actions happening to begin with depends on whether the environment initially offers these actions, bnt not on what it offers thereafter.

We now give the semantics of *PCSP*. All definitions are for any $\mathcal{F}$-set A and trace $y \in \Omega$.

7.1 STOP

As always, the simplest process is *STOP*, which deadlocks no matter what the environment offers (and is therefore constant with respect to y):

$$(\!|STOP|\!)\rho\ (A, y) \;\widehat{=}\; I_A\langle\tau\rangle^\omega.$$

7.2 Prefixing

If the environment offers an $a \in \Sigma$, the probability of $a \to P$ performing a set A is the probability of P performing $prefix_a^{-1}A$, which depends on the second and further actions offered by the environment. If the environment does not offer a, then $a \to P$ behaves like *STOP*. We therefore define

$$\begin{aligned}&(\!|a \to P|\!)\rho\ (A, y)\\ &\quad\widehat{=}\; I_{S\langle a\rangle}(y)\ (\!|P|\!)\rho\ (prefix_a^{-1}A, y/1) + I_{S\langle a\rangle^c}(y)\ (\!|STOP|\!)\rho\ (A, y).\end{aligned}$$

It follows from lemma 2.1.9 that this defines a cpm.

7.3 External Choice

For a set of visible actions E define $e : E \to P_e$ to be the process which, when offered an action e in E, performs e with probability 1 and then behaves like P_e conditioned on the second and fnrther steps of the environment. When offered an action ontside E the process deadlocks.

$$\begin{aligned}&(\!|e : E \to P_e|\!)\rho\ (A, y)\\ &\quad\widehat{=}\; \sum_{e\in E} I_{S\langle e\rangle}(y)\ (\!|P_e|\!)\rho\ (\rho refix_e^{-1}A, y/1) + \sum_{e\notin E} I_{S\langle e\rangle}(y)\ (\!|STOP|\!)\rho\ (A, y).\end{aligned}$$

It follows from lemma 2.1.10 that this is a cpm. Note that $STOP$ and $a \rightarrow P$ are special cases of external choice with $E = \emptyset$ and $E = \{a\}$ respectively.

We adopt a special notation for communication. Let $c.v$ denote an action with two components. The first component, c, is the name of the channel on which the communication takes place. The second component, v, is the value of the message which passes. A process which first outputs v on channel c and then behaves as P is defined

$$c!v \rightarrow P \;\hat{=}\; c.v \rightarrow P.$$

A process which is initially prepared to input any value commnnicable on channel c is defined

$$c?x \rightarrow P_x \;\hat{=}\; d : \{e \mid chan(e) = c\} \rightarrow P_{msg(d)}.$$

where $chan(c.v) = c$ and $msg(c.v) = v$.

7.4 Probabilistic choice

As before we write $P \;{}_p\!\sqcap\; Q$ to denote probabilistic choice. Its semantics in terms of conditional probabilities hardly needs explanation: it equals the weighted sum of the conditional probabilities of the component processes.

$$(\!|P \;{}_p\!\sqcap\; Q|\!)\rho(A, y) \;\hat{=}\; p\,(\!|P|\!)\rho(A, y) + (1-p)\,(\!|Q|\!)\rho(A, y).$$

For fixed y, $(\!|P \;{}_p\!\sqcap\; Q|\!)$ is a weighted sum of probability measures, and thus itself a probability measure by lemma 2.1.4. For fixed A it is a sum of random variables and thus itself a random variable. Therefore it is a cpm.

All the laws which hold in $PCSP_0$ also hold in $PCSP$. Additionally there is a law which relates probabilistic and external choice:

Lemma 7.4.1

L1 $P \;{}_p\!\sqcap\; P \equiv P.$

L2 $P \;{}_p\!\sqcap\; Q \equiv Q \;{}_{1-p}\!\sqcap\; P.$

L3 $P \;{}_1\!\sqcap\; Q \equiv P.$

L4 $(P \;{}_{p/(1-q)}\!\sqcap\; Q) \;{}_{1-q}\!\sqcap\; R \equiv (R \;{}_{q/(1-p)}\!\sqcap\; Q) \;{}_{1-p}\!\sqcap\; P.$

Probabilistic choice distributes over external choice and prefixing:

L5 $e : E \to (P_e \ {}_p\sqcap \ Q_e) \ \equiv \ (e : E \to P_e) \ {}_p\sqcap \ (e : E \to Q_e)$.

L6 $a \to (P \ {}_p\sqcap \ Q) \ \equiv \ (a \to P) \ {}_p\sqcap \ (a \to Q)$.

□

Proof Laws 1 to 3 are obvious. The proof of associativity (law 4) is the same as in $PCSP_0$ if we snbstitute cpms for probability measures. For law 5 note that

$$
\begin{aligned}
&(\!|e : E \to (P_e \ {}_p\sqcap \ Q_e)|\!) \ (A, y) \\
&= \sum_{e \in E} I_{S\langle e\rangle}(y) \ (\!|P_e \ {}_p\sqcap \ Q_e|\!) \ (prefix_e^{-1} A, y/1) + \sum_{e \notin E} I_{S\langle e\rangle}(y) \ (\!|STOP|\!)(A, y) \\
&= \ p \ (\sum_{e \in E} I_{S\langle e\rangle}(y) \ (\!|P_e|\!) \ (prefix_e^{-1} A, y/1) + \sum_{e \notin E} I_{S\langle e\rangle}(y) \ (\!|STOP|\!) \ (A, y)) \\
&\quad + (1-p) \\
&\quad \ (\sum_{e \in E} I_{S\langle e\rangle}(y) \ (\!|Q_e|\!) \ (prefix_e^{-1} A, y/1) + \sum_{e \notin E} I_{S\langle e\rangle}(y) \ (\!|STOP|\!) \ (A, y)) \\
&= \ (\!|(e : E \to P_e) \ {}_p\sqcap \ (e : E \to Q_e)|\!) \ (A, y)
\end{aligned}
$$

Law 6 is a special case of law 5. □

7.5 General Choice

The general choice operator in *CSP* denotes external choice between processes rather than actions. The same is true for general choice in *PCSP*. We write $P \ {}_S\Box \ Q$ for a process which behaves like P when offered a trace in S, and like Q when offered a trace in S^c:

$$(\!|P \ {}_S\Box \ Q|\!)\rho \ (A, y) \ \ \hat{=} \ \ I_S(y) \ (\!|P|\!)\rho \ (A, y) + I_{S^c}(y) \ (\!|Q|\!)\rho \ (A, y)$$

where S must be such that for any trace $t \in \Sigma_\downarrow^*$

$$y \in S \cap S(t) \wedge z \in S^c \cap S(t) \ \ \Rightarrow \ \ (\!|P|\!) \ (S(t), y) = (\!|Q|\!) \ (S(t), z).$$

The fact that tbis is a cpm follows from lemma 2.1.10.

To see why not all sets S are admissible suppose $a \to P \ {}_{S\langle a,b\rangle}\Box \ STOP$ to be a valid process definition. Then $(\!|a \to P \ {}_{S\langle a,b\rangle}\Box \ STOP|\!) \ (S\langle a\rangle, z)$ would be 1 if $z_1 = b$ and 0 if $z_1 \neq b$. This violates the rule that an action, a, must not depend on anything that might happen afterwards, b.

General choice satisfies most of the laws which we would expect it to, bnt in one respect it is different from other models of *CSP*. In the failures-divergences

model [Hoa85] the choice between identical initial actions degenerates into non-deterministic choice:

$$\begin{aligned} a \to P \,\Box\, a \to Q &\equiv (a \to P) \sqcap (a \to Q) \\ &\equiv a \to (P \sqcap Q). \end{aligned}$$

In the probabilistic model, a process with a general choice between two branches beginning with a will also do an a first, but then the choice of P or Q depends on the second step of the environment:

$$a \to P \;{}_S\Box\; a \to Q \equiv a \to (P \;{}_{prefix_a^{-1}S}\Box\; Q)$$

The other laws for general choice are similar to those for probabilistic choice. This is to be expected since the semantics of both operators are defined in terms of sums of cpm's.

Lemma 7.5.1

L1 $P \;{}_S\Box\; P \equiv P.$

L2 $P \;{}_S\Box\; Q \equiv Q \;{}_{S^c}\Box\; P.$

L3 $P \;{}_\Omega\Box\; Q \equiv P.$

L4 $(P \;{}_{S_1}\Box\; Q) \;{}_{S_2}\Box\; R \equiv P \;{}_{S_1 \cap S_2}\Box\; (Q \;{}_{S_2}\Box\; R).$

Probabilistic and general choice distribute over each other.

L5 $P \;{}_S\Box\; (Q \;{}_p\sqcap\; R) \equiv (P \;{}_S\Box\; Q) \;{}_p\sqcap\; (P \;{}_S\Box\; R).$

L6 $(P \;{}_S\Box\; Q) \;{}_p\sqcap\; R \equiv (P \;{}_p\sqcap\; R) \;{}_S\Box\; (Q \;{}_p\sqcap\; R).$

The next law is the probabilistic analogue of the *CSP*-law $P \,\Box\, STOP \equiv P$.

L7 $P \;{}_{(S(\tau))^c}\Box\; STOP \equiv P.$

L8 If $S \supseteq \{u | u_0 \in E - D\}$ and $S^c \supseteq \{u | u_0 \in D - E\}$ then

$$(e : E \to P_e) \;{}_S\Box\; (d : D \to Q_d) \equiv d : E \cup D \to R_d$$

where, for $S' = prefix_d^{-1}S$, $R_d \equiv \begin{cases} P_d \;{}_{S'}\Box\; Q_d & \text{if } d \in E \cap D \\ P_d & \text{if } d \in E - D \\ Q_d & \text{if } d \in D - E. \end{cases}$

As a corollary to law 8 we can write

L9 $a \to P \ {}_S\Box \ a \to Q \equiv a \to (P \ {}_{prefix_a^{-1}S}\Box \ Q)$.

□

Proof Law 1 follows directly from the definition:

$$I_S(y)\,(\!|P|\!)\,(A,y) + I_{S^c}(y)\,(\!|P|\!)\,(A,y) = (\!|P|\!)\,(A,y)$$

Symmetry (law 2) is obvious, as is law 3. In the proofs of associativity (law 4) and distributivity (law 5) we suppress the arguments (A, y) which are carried through the whole proof unchanged.

$$\begin{aligned}(\!|(P \ {}_{S_1}\Box \ Q) \ {}_{S_2}\Box \ R|\!) &= I_{S_2}\,(I_{S_1}\,(\!|P|\!) + I_{S_1^c}\,(\!|Q|\!)) + I_{S_2^c}\,(\!|R|\!)\\ &= I_{S_1\cap S_2}\,(\!|P|\!) + I_{S_1^c\cap S_2}\,(\!|Q|\!) + I_{S_2^c}\,(\!|R|\!)\\ &= (\!|P \ {}_{S_1\cap S_2}\Box \ (Q \ {}_{S_2}\Box \ R)|\!).\end{aligned}$$

To prove that general choice distributes over non-deterministic choice simply expand and regroup the terms: for law 5 we get

$$\begin{aligned}(\!|P \ {}_S\Box \ (Q \ {}_p\sqcap \ R)|\!) &= I_S\,(\!|P|\!) + I_{S^c}\,(p\,(\!|Q|\!) + (1-p)\,(\!|R|\!))\\ &= p\,(I_S\,(\!|P|\!) + I_{S^c}\,(\!|Q|\!)) + (1-p)\,(I_S(\!|P|\!) + I_{S^c}(\!|R|\!))\\ &= (\!|(P \ {}_S\Box \ Q) \ {}_p\sqcap \ (P \ {}_S\Box \ R)|\!).\end{aligned}$$

Similarly for law 6. To prove law 7 recall that for any process P and for all $y \in \Omega$ we have $(\!|P|\!)\,(\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}, y) = 1$. Accordingly, $(\!|P|\!)\,(\{\langle\tau\rangle^\omega\}, \langle\tau\rangle^\omega) = 1$ for any P, i.e. no process can do anything when the environment offers it $\langle\tau\rangle^\omega$. So

$$(\!|P|\!)\,(A, \langle\tau\rangle^\omega) = (\!|STOP|\!)\,(A, \langle\tau\rangle^\omega).$$

Law 7 follows. For law 8 we expand

$$\begin{aligned}&(\!|(e : E \to P_e) \ {}_S\Box \ (d : D \to Q_d)|\!)\,(A, y)\\ &= I_S(y)\,(\sum_{e\in E} I_{S\langle e\rangle}(y)\,(\!|P_e|\!)\,(prefix_e^{-1}A, y/1) + \sum_{e\notin E} I_{S\langle e\rangle}(y)\,(\!|STOP|\!)\,(A, y))\\ &\quad + I_{S^c}(y)(\sum_{d\in D} I_{S\langle d\rangle}(y)\,(\!|Q_d|\!)\,(prefix_d^{-1}A, y/1) + \sum_{d\notin D} I_{S\langle d\rangle}(y)\,(\!|STOP|\!)\,(A, y))\\ &= \sum_{d\in E\cap D} I_{S\langle d\rangle}(y)\,(I_S(y)\,(\!|P_d|\!)\,(prefix_d^{-1}A, y/1) + I_{S^c}(y)\,(\!|Q_d|\!)\,(prefix_d^{-1}A, y/1))\\ &\quad + \sum_{d\in E-D} I_{S\langle d\rangle}(y)\,(I_S(y)\,(\!|P_d|\!)\,(prefix_d^{-1}A, y/1) + I_{S^c}(y)\,(\!|STOP|\!)\,(A, y))\\ &\quad + \sum_{d\in D-E} I_{S\langle d\rangle}(y)\,(I_S(y)\,(\!|STOP|\!)\,(A, y) + I_{S^c}(y)\,(\!|Q_d|\!)\,(prefix_d^{-1}A, y/1))\\ &\quad + \sum_{d\notin E\cup D} I_{S\langle d\rangle}(y)\,(\!|STOP|\!)\,(A, y).\end{aligned}$$

Since $S \supseteq \{u \mid u_0 \in E - D\}$ and $S^c \supseteq \{u \mid u_0 \in D - E\}$ this simplifies to

$$\begin{aligned}
&\sum_{d \in E \cap D} I_{S(d)}(y)\,(I_S(y)\,(\!|P_d|\!)\,(prefix_d^{-1}A, y/1) + I_{S^c}(y)\,(\!|Q_d|\!)\,(prefix_d^{-1}A, y/1)) \\
&\quad + \sum_{d \in E - D} I_{S(d)}(y)\,(\!|P_d|\!)\,(prefix_d^{-1}A, y/1) \\
&\quad + \sum_{d \in D - E} I_{S(d)}(y)\,(\!|Q_d|\!)\,(prefix_d^{-1}A, y/1)) \\
&\quad + \sum_{d \notin E \cup D} I_{S(d)}(y)\,(\!|STOP|\!)\,(A, y) \\
&= (\!|d : E \cup D \to R_d|\!)\,(A, y)
\end{aligned}$$

where R_d is the same as in law 8. □

If $\{S_i\}_{0 \le i < n}$ is a finite partition of Ω we write

$$\Box_{S_i} P_i$$

for the prefix form of general choice. Also, if all the branches of the choice are guarded and the sets on which they are conditioned coincide with the guards, we can omit the sets: for example $a \to P \,\Box\, b \to P \equiv a \to P \;{}_{S(a)}\Box\; b \to P$.

7.6 Simple Parallel Composition

Two processes which operate in lockstep parallel must synchronise with each other and with their common environment at every step. Thus if the environment offers a trace z to the parallel system $P \parallel Q$, then the components behave as P given z and Q given z, and interact in the same way as the corresponding probability measures in the model $PCSP_0$. That is we define simple parallel composition as a transformation of cpm's based on the function *par*:

$$(\!|P \parallel Q|\!)\rho\,(A, z) \;\hat{=}\; \int (\!|P|\!)\rho\,((par^{-1}A)_y, z)(\!|Q|\!)\rho\,(dy, z).$$

We have already shown in chapter 3 that *par* is measurable $(\mathcal{F} \times \mathcal{F})/\mathcal{F}$. Therefore by lemmas 2.1.9 and 2.1.8 the above defines a cpm.

Not surprisingly all the laws for simple parallel composition which hold in $PCSP_0$ also hold in $PCSP$, but additionally there is a law (law 8) which relates parallel composition to external choice.

Lemma 7.6.1

L1 $P \parallel Q \equiv Q \parallel P$.

L2 $P \parallel (Q \parallel R) \equiv (P \parallel Q) \parallel R.$

L3 $P \parallel STOP \equiv STOP.$

L4 $(P \,{}_p\!\sqcap\, Q) \parallel R \equiv P \parallel R \,{}_p\!\sqcap\, Q \parallel R.$

L5 $P \parallel (Q \,{}_p\!\sqcap\, R) \equiv P \parallel Q \,{}_p\!\sqcap\, P \parallel R.$

L6 $(a \rightarrow P) \parallel (a \rightarrow Q) \equiv a \rightarrow (P \parallel Q).$

L7 $a \neq b \Rightarrow (a \rightarrow P) \parallel (b \rightarrow Q) \equiv STOP.$

L8 $(e : E \rightarrow P_e) \parallel (g : G \rightarrow Q_d) \equiv d : E \cap G \rightarrow (P_d \parallel Q_d).$

□

We will show in the next section that simple parallel composition is a special case of alphabetised parallel composition. So the laws for simple parallel composition need not be proved separately.

7.7 Alphabetised Parallel Composition

Let B and C be two sets of actions such that $\tau \in B \subseteq \Sigma_\tau$, $\tau \in C \subseteq \Sigma_\tau$. We write $P \,{}_B\|_C\, Q$ to denote the parallel composition of two processes P and Q such that P can perform actions in B and Q can perform actions in C and P and Q synchronise on actions in the intersection $B \cap C$. The environment of this system participates in every action and can be thought of as a scheduler or adversary. If the environment offers a trace $z \in \Sigma_\tau$ to the system, then the component P is affected only by the steps in z which are elements of B. So it behaves as P given $z \restriction B$, provided $z \restriction B$ is infinite. If it is finite then P cannot do anything beyond $z \restriction B$, i.e. it behaves as P given $(z \restriction B)\langle\tau\rangle^\omega$. Apart from the fact that the sequence offered by the environment determines how the actions performed by different components are to be interleaved we want parallel composition to work in the same way as in chapter 6, that is the probability that the system performs a sequence of actions should be the product of the probabilities with which the components take part. We therefore define parallel composition as a transformation of the product of the component cpm's, based on a function which merges two sequences x and y as far as possible in accordance with the sequence offered by the environment.

$$(\!|P \,{}_B\|_C\, Q|\!)\rho\,(A, z) \;\widehat{=}\; \int (\!|P|\!)\rho\,((cpar^{-1}_{B,C,z}A)_y, z \restriction B)\,(\!|Q|\!)\rho\,(dy, z \restriction C)$$

where

$$cpar_{B,C,z} : \Omega \times \Omega \to \Omega$$
$$\forall x, y \in \Omega \cdot \quad cpar_{B,C,z}(x,y) \;=\; merge_{B,\Sigma_\tau}(x, merge_{C,\Sigma_\tau}(y,z))$$

(Recall that $\restriction$ is the restriction function which adds a tail of τ's where $\restriction$ produces a finite trace.)

To show that the above transformation defines a cpm we only need to prove that *cpar* is a measurable function.

Lemma 7.7.1 The function $cpar_{B,C,z}$ is measurable $\mathcal{F}\times\mathcal{F}/\mathcal{F}$. □

Proof We first expand the definition of $cpar_{B,C,z}$:

$$\begin{aligned} & cpar_{B,C,z}(x,y) \\ &= \begin{cases} merge_{B,\Sigma_\tau}(x,z) & \text{if } z\restriction C \le y \\ merge_{B,\Sigma_\tau}(x,(z\restriction n)\langle\tau\rangle^\omega) & \text{if } (z\restriction n)\restriction C < y \wedge (z\restriction n+1)\restriction C \not< y \end{cases} \\ &= \begin{cases} z & \text{if } z\restriction B \le x \wedge z\restriction C \le y \\ (z\restriction n)\langle\tau\rangle^\omega & \text{if } (z\restriction n)\restriction B < x \wedge (z\restriction n)\restriction C < y \\ & \quad \wedge ((z\restriction n+1)\restriction B \not< x \vee (z\restriction n+1)\restriction C \not< y) \end{cases} \end{aligned}$$

If t is a finite trace such that $t \not< z$ and $\neg\exists\, n \cdot t < (z\restriction n)\langle\tau\rangle^\omega$ then $cpar^{-1}_{B,C,z}S(t) = \emptyset$. Otherwise, if t is τ-free then we must have $t = z\restriction\#t$ and

$$cpar^{-1}_{B,C,z}S(t) \;=\; S(t\restriction B)\times S(t\restriction C).$$

If t contains a tail of τ's then $\exists\, t' \in \Sigma^*$ such that $t' < t < t'\langle\tau\rangle^\omega$ and

$$cpar^{-1}_{B,C,z}S(t) \;=\; cpar^{-1}_{B,C,z}S(t') - \bigcup_{e\neq\tau} cpar^{-1}_{B,C,z}S(t'\langle e\rangle)$$

which, as a difference of measurable sets, is also measurable. So $cpar_{B,C,z}$ is measurable. □

Lemma 7.7.2 Simple parallel composition is equivalent to alphabetised parallel composition which is synchronised on all actions: $P \parallel Q \;\equiv\; P \;{}_{\Sigma_\tau}\|_{\Sigma_\tau}\; Q$. □

Proof Note that $\forall x, y \in \Omega$

$$\begin{aligned} & cpar_{\Sigma_\tau,\Sigma_\tau,z}(x,y) \\ &= \begin{cases} z & \text{if } x = y = z \\ (z\restriction n)\langle\tau\rangle^\omega & \text{if } z\restriction n = x\restriction n = y\restriction n \wedge (z_n \neq x_n \vee z_n \neq y_n). \end{cases} \\ &= par(par(x,y),z) \end{aligned}$$

By definition

$$(\!|P \parallel Q|\!)(A, z) \;=\; \int (\!|P|\!)((par^{-1}A)_{\mathbf{y}}, z)\,(\!|Q|\!)(dy, z).$$

The support of the product measure is $T \times T$ where $T \triangleq \{z\} \cup \bigcup_n \{(z\restriction n)\langle\tau\rangle^\omega\}$. But

$$\begin{array}{l} x, y \in T \;\Rightarrow \\ \quad par(x, y) \;=\; par(par(x, y), z) \;=\; cpar_{\Sigma_\tau, \Sigma_\tau, z}(x, y). \end{array}$$

Therefore

$$\begin{array}{rcl} (\!|P \parallel Q|\!)(A, z) & = & \int (\!|P|\!)((cpar^{-1}_{\Sigma_\tau, \Sigma_{\tau,z}} A)_{\mathbf{y}}, z)\,(\!|Q|\!)(dy, z) \\ & = & (\!|P_{\,\Sigma_\tau} \|_{\Sigma_\tau} Q|\!)(A, z). \end{array}$$

□

Lemma 7.7.3

L1 $P \,{}_B\|_C\, Q \;\equiv\; Q \,{}_B\|_C\, P.$

L2 $(P \,{}_B\|_C\, Q) \,{}_{B \cup C}\|_D\, R \;\equiv\; P \,{}_B\|_{C \cup D}\, (Q \,{}_C\|_D\, R).$

L3 $B \subseteq C \Rightarrow\; P \,{}_B\|_C\, STOP \;\equiv\; STOP.$

L4 $(P \,{}_{\mathbf{p}}\sqcap\; Q) \,{}_B\|_C\, R \;\equiv\; P \,{}_B\|_C\, R \,{}_{\mathbf{p}}\sqcap\; Q \,{}_B\|_C\, R.$

L5 $P \,{}_B\|_C\, (Q \,{}_{\mathbf{r}}\sqcap\; R) \;\equiv\; P \,{}_B\|_C\, Q \,{}_{\mathbf{p}}\sqcap\; P \,{}_B\|_C\, R.$

L6 $a \in B \cap C \Rightarrow$
$(a \to P) \,{}_B\|_C\, (a \to Q) \;\equiv\; a \to (P \,{}_B\|_C\, Q).$

L7 $a \in B \cap C, b \in B - C \Rightarrow$
$(b \to P) \,{}_B\|_C\, (a \to Q) \;\equiv\; b \to (P \,{}_B\|_C\, a \to Q).$

L8 $a, b \in B \cap C, a \neq b \Rightarrow$
$(b \to P) \,{}_B\|_C\, (a \to Q) \;\equiv\; STOP.$

L9 $b \in B - C, c \in C - B \Rightarrow$
$(b \to P) \,{}_B\|_C\, (c \to Q) \;\equiv\; b \to (P \,{}_B\|_C\, c \to Q) \,{}_{S(b)}\Box\, c \to ((b \to P) \,{}_B\|_C\, Q).$

The last four laws are generalised by the following:

L10 Let E and D be sets of visible actions such that $E \subset B$ and $D \subset C$. Then

$$(e : E \to P_e) \;{}_B\|_C\; (d : D \to Q_d) \;\equiv\; g : G \to P' \;{}_B\|_C\; Q'$$

where $G = (E \cap D) \cup (E - C) \cup (D - B)$
$P' = P_g$ if $g \in E$, P otherwise
$Q' = Q_g$ if $g \in D$, Q otherwise .

□

Proof Symmetry follows from the symmetry of *cpar* and Fubini's theorem. For associativity we first prove that

$$(cpar_{B,C,u}, id)\,;\, cpar_{B\cup C,D,u} \;=\; (id, cpar_{C,D,u})\,;\, cpar_{B,C\cup D,u}.$$

Expanding the left hand side we get $\forall x, y, z \in \Omega$

$$\begin{aligned}
&cpar_{B\cup C,D,u}(cpar_{B,C,u}, id)(x,y,z)\\
&= \begin{cases} cpar_{B\cup C,D,u}(u,z) & \text{if } u\restriction B \le x \wedge u\restriction C \le y\\ cpar_{B\cup C,D,u}((u\restriction n)\langle r\rangle^\omega, z) & \text{if } (u\restriction n)\restriction B < x \wedge (u\restriction n)\restriction C < y\\ & \quad \wedge ((u\restriction n+1)\restriction B \not\le x \vee (u\restriction n+1)\restriction C \not\le y)\end{cases}\\
&= \begin{cases} u & \text{if } u\restriction B \le x \wedge u\restriction C \le y \wedge u\restriction D \le z\\ (u\restriction n)\langle r\rangle^\omega & \text{if } (u\restriction n)\restriction B < x \wedge (u\restriction n)\restriction C < y \wedge (u\restriction n)\restriction D < z\\ & \quad \wedge ((u\restriction n+1)\restriction B \not< x \vee (u\restriction n+1)\restriction C \not< y\\ & \quad \vee (u\restriction n+1)\restriction D \not< z).\end{cases}
\end{aligned}$$

Because of the symmetry of *cpar* we can write

$$cpar_{B,C\cup D,u}(id, cpar_{C,D,u})(x,y,z) \;=\; cpar_{C\cup D,B,u}(cpar_{C,D,u}, id)(y,z,x)$$

and use the above formula to expand it and show equality. It follows that

$$\begin{aligned}
(cpar^{-1}_{B,C,u}(cpar^{-1}_{B\cup C,D,u}A)_z)_x &= (((cpar_{B,C,u}, id)\,;\, cpar_{B\cup C,D,u})^{-1}A)_{x,z}\\
&= (((id, cpar_{C,D,u})\,;\, cpar_{B,C\cup D,u})^{-1}A)_{x,z}\\
&= (cpar^{-1}_{C,D,u}(cpar^{-1}_{B,C\cup D,u}A)_x)_z.
\end{aligned}$$

This together with Fubini's theorem allows us to prove associativity:

$$\begin{aligned}
&(\!|(P \;{}_B\|_C\; Q) \;{}_{B\cup C}\|_D\; R|\!)(A,u)\\
&= \int (\!|P \;{}_B\|_C\; Q|\!)((cpar^{-1}_{B\cup C,D,u}A)_z, u\restriction B\cup C)\;(\!|R|\!)(dz, u\restriction D)\\
&= \int\int (\!|Q|\!)((cpar^{-1}_{B,C,u}(cpar^{-1}_{B\cup C,D,u}A)_z)_x, u\restriction C)\;(\!|P|\!)(dx, u\restriction B)\;(\!|R|\!)(dz, u\restriction D)\\
&= \int\int (\!|Q|\!)((cpar^{-1}_{C,D,u}(cpar^{-1}_{B,C\cup D,u}A)_x)_z, u\restriction C)\;(\!|R|\!)(dz, u\restriction D)\;(\!|P|\!)(dx, u\restriction B)\\
&= \int (\!|Q \;{}_C\|_D\; R|\!)((cpar^{-1}_{B,C\cup D,u}A)_x, u\restriction C\cup D)\;(\!|P|\!)(dx, u\restriction B)\\
&= (\!|P \;{}_B\|_{C\cup D}\; (Q \;{}_C\|_D\; R)|\!)(A,u).
\end{aligned}$$

For law 3 note that

$$\begin{aligned}
(\!|P\ {}_B\|_C\ STOP|\!)\,(A,z) &= \int (\!|P|\!)\,((cpar^{-1}_{B,C,z}A)_y, z\restriction B)\ (\!|STOP|\!)\,(dy, z\restriction C)\\
&= (\!|P|\!)\,((cpar^{-1}_{B,C,z}A)_{\langle\tau\rangle^\omega}, z\restriction B)\\
&\quad \text{since } \forall z \cdot (\!|STOP|\!)\,(\{\langle\tau\rangle^\omega\}, z) = 1\\
&= \begin{cases} (\!|P|\!)\,(\Omega, z\restriction B) & \text{if } \langle\tau\rangle^\omega \in A\\ (\!|P|\!)\,(\emptyset, z\restriction B) & \text{otherwise}\end{cases}\\
&\quad \text{since } epar_{B,C,z}(x, \langle\tau\rangle^\omega) = \langle\tau\rangle^\omega \text{ if } B \subseteq C\\
&= (\!|STOP|\!)\,(A, z\restriction B)\\
&= (\!|STOP|\!)\,(A, z).
\end{aligned}$$

The proofs of distributivity of alphabetised parallel composition over probabilistic choice follow along the same lines as the proofs of the corresponding laws for simple parallel composition in $PCSP_0$. Since laws 6 – 9 are special cases of law 10 the latter is the only one which remains to be proven. Let $(\!|P|\!) = (\!|e : E \to P_e|\!)$ and let $(\!|Q|\!) = (\!|d : D \to Q_d|\!)$. By definition

$$(\!|P\ {}_B\|_C\ Q|\!)(A,z) = \int (\!|P|\!)\,((cpar^{-1}_{B,C,z}A)_y, z\restriction B)\ (\!|Q|\!)(dy, z\restriction C).$$

If $z_0 = e \in E \cap D$ then $(z\restriction B)_0 = e$ and $(z\restriction C)_0 = e$. Therefore in this case

$$\begin{aligned}
&(\!|P\ {}_B\|_C\ Q|\!)(A,z)\\
&= \int (\!|e \to P_e|\!)\,((cpar^{-1}_{B,C,z}A)_w, z\restriction B)\ (\!|e \to Q_e|\!)(dw, z\restriction C)\\
&= \int (\!|P_e|\!)\,(prefix_e^{-1}(cpar^{-1}_{B,C,z}A)_{\langle e\rangle w}, (z\restriction B)/1)\ (\!|Q_e|\!)(dw, (z\restriction C)/1)\\
&= \int (\!|P_e|\!)\,(cpar_{B,C,z/1}(prefix_e^{-1}A)_w, (z/1)\restriction B)\ (\!|Q_e|\!)(dw, (z/1)\restriction C)\\
&= (\!|P_e\ {}_B\|_C\ Q_e|\!)\,(prefix_e^{-1}A, z/1)\\
&= (\!|e \to (P_e\ {}_B\|_C\ Q_e)|\!)\,(A,z).
\end{aligned}$$

If $z_0 = e \in E - C$ then $(z\restriction B)_0 = e$ and $z\restriction C = (z/1)\restriction C$. Therefore

$$\begin{aligned}
&(\!|P\ {}_B\|_C\ Q|\!)\,(A,z)\\
&= \int (\!|e \to P_e|\!)\,((cpar^{-1}_{B,C,z}A)_y, (z\restriction B))\ (\!|Q|\!)(dy, z\restriction C)\\
&= \int (\!|P_e|\!)\,(prefix_e^{-1}(cpar^{-1}_{B,C,z}A)_y, (z\restriction B)/1)\ (\!|Q|\!)(dy, z\restriction C)\\
&= \int (\!|P_e|\!)\,(cpar_{B,C,z/1}(prefix_e^{-1}A)_y, (z/1)\restriction B)\ (\!|Q|\!)(dy, (z/1)\restriction C)\\
&= (\!|P_e\ {}_B\|_C\ Q|\!)\,(prefix_e^{-1}A, z/1)\\
&= (\!|e \to (P_e\ {}_B\|_C\ Q)|\!)\,(A,z).
\end{aligned}$$

Similarly if $z_0 = e \in D - B$ then

$$(\!|P\ {}_B\|_C\ Q|\!)(A,z) \;=\; (\!|e \to (P\ {}_B\|_C\ Q_e)|\!)(A,z).$$

If $z_0 \in (B-E) \cap C$ then $(z\!\upharpoonright\! B)_0 = z_0$ and $(z\!\upharpoonright\! C)_0 = z_0$. So

$$(\!|P\ {}_B\|_C\ Q|\!)(A,z) \;=\; \int (\!|STOP|\!)((cpar^{-1}_{B,C,z}A)_y, (z\!\upharpoonright\! B))\ (\!|Q|\!)(dy, z\!\upharpoonright\! C).$$

Also $z_0 \in (B-E) \cap C \Rightarrow cpar_{B,C,z}(\langle\tau\rangle^\omega, y) = \langle\tau\rangle^\omega$ for all y. Therefore it follows that $cpar^{-1}_{B,C,z}A \supseteq \{\langle\tau\rangle^\omega\} \times \Omega \Leftrightarrow \langle\tau\rangle^\omega \in A$ and

$$(\!|P\ {}_B\|_C\ Q|\!)(A,z) \;=\; (\!|STOP|\!)(A,z).$$

The same is true if $z_0 \in (C-D) \cap B$. Finally, if $z_0 \in (D \cup C)^c$ then

$$\begin{aligned}(\!|P\ {}_B\|_C\ Q|\!)(A,z) &= \int (\!|STOP|\!)((cpar^{-1}_{B,C,z}A)_y, (z\!\upharpoonright\! B))\ (\!|STOP|\!)(dy, z\!\upharpoonright\! C)\\ &= (\!|STOP|\!)(A,z)\end{aligned}$$

since $\{\langle\tau\rangle^\omega\} \times \{\langle\tau\rangle^\omega\} \in cpar^{-1}_{B,C,z}A \Leftrightarrow \langle\tau\rangle^\omega \in A$. Drawing all these cases together, we get

$$\begin{aligned}(\!|P\ {}_B\|_C\ Q|\!)(A,z) \;=\;& \sum_{e \in E \cap D} I_{S(e)}(z)(\!|e \to (P_e\ {}_B\|_C\ Q_e)|\!)(A,z)\\ &+ \sum_{e \in E-C} I_{S(e)}(z)(\!|e \to (P_e\ {}_B\|_C\ Q)|\!)(A,z)\\ &+ \sum_{e \in D-B} I_{S(e)}(z)(\!|e \to (P\ {}_B\|_C\ Q_e)|\!)(A,z)\\ &+ \sum_{e \in \Sigma-G} I_{S(e)}(z)(\!|STOP|\!)(A,z)\end{aligned}$$

where $G = (E \cap D) \cup (E-C) \cup (D-B)$. Thus as required

$$(\!|P\ {}_B\|_C\ Q|\!)(A,z) \;=\; (\!|g : G \to (P'\ {}_B\|_C\ Q')|\!)(A,z)$$

where P', Q' as defined in Law 10. □

We write $\|_{B_i}\ P_i$ for the prefix form of parallel composition. Each component process P_i may perform only actions which are in the corresponding set B_i. The behaviour of the whole system is the pairwise evaluation of the parallel composition components (by associativity).

7.8 Relabelling

Recall that given a cpm P and functions f and g a cpm P' can be defined as a transformation of P by setting $P'(A,z) \triangleq P(f^{-1}A, g\ z)$. For the prefixing operator we use $g\ z = \mathit{prefix}_e^{-1} z$. For parallel composition we use $g\ z = (z\restriction B, z\restriction C)$ which is an element of $\mathit{cpar}^{-1}_{B,C,z} z$. An attempt to define hiding or sequential composition similarly fails because there is no sensible way of selecting an element of the inverses of *hide* and *seq*.

$$\begin{aligned} \mathit{hide}_B^{-1} z &= \{u \mid u\restriction B = z\} \\ \mathit{seq}^{-1} z &= \bigcup_n S((z\restriction n)\langle\surd\rangle) \times \{z/n\} \cup \{z\} \times \Omega. \end{aligned}$$

However, we can define relabelliug if we restrict ourselves to injective relabelling functions. Let $f : \Sigma \to \Sigma$ be such a function, which is lifted to sequences in the usual way. Then

$$(\!|f(P)|\!)\rho\ (A,z) \triangleq (\!|P|\!)\rho\ (f^{-1}A, f^{-1}z)$$

defines a cpm.

It is easily checked that relabelling satisfies the following laws:

Lemma 7.8.1

L1 $f(STOP) \equiv STOP.$

L2 $f(g(P)) \equiv (g\,;f)P.$

L3 $f(e : E \to P_e) \equiv e : f(E) \to f(P)_e.$

L4 $f(P\ {}_p\sqcap\ Q) \equiv f(P)\ {}_p\sqcap\ f(Q).$

L5 $f(P\ {}_B\|_C\ Q) \equiv f(P)\ {}_{f(B)}\|_{f(C)}\ f(Q).$

□

7.9 Conditional

We use an infix-notation for conditionals. For a boolean expression b and processes P, Q define

$$(\!|P \lhd b \rhd Q|\!) \triangleq \begin{cases} (\!|P|\!) & \text{if } b = \mathit{true} \\ (\!|Q|\!) & \text{otherwise .} \end{cases}$$

7.10 Recursion

To give a semantics to recursion in terms of cpm's we use the same approach as in chapter 4: we define the semantics of a recursive expression to be the fixed point of an eqnation and use a metric on the space of cpm's and the Banach Fixed Point Theorem to establish conditions of well-definedness.

Let P be a term possibly containing the free variable X. As before, we write $\mu X \bullet P$ to denote a process that behaves as P with X representing a recursive invocation of the process. To define its semantics we regard $(\!|P|\!)\rho$ as a function of the cpm to be bound to the variable X in P:

Definition 7.10.1 If P is a $PCSP$ term possibly containing the free variable X then $M(X,P)\rho \triangleq \lambda\, Y \cdot (\!|P|\!)\rho[Y/X]$. □

We can then define

$$(\!|\mu X \bullet P|\!)\rho \quad \triangleq \quad \text{the unique fixed point of the mapping } M(X,P)\rho.$$

We use two metrics which are closely related to the ones we used for the semantics of $PCSP_0$. Given two cpm's P and Q define

$$\delta(P,Q) \quad \triangleq \quad \sup_{z \in \Sigma^\omega} \sum_{n=0}^{\infty} \frac{1}{2^n} |P(S(z\restriction n), z) - Q(S(z\restriction n), z)|$$

and

$$\delta'(P,Q) \quad \triangleq \quad \inf\{2^{-n} \mid \forall\, z \in \Sigma^\omega \cdot P(S(z\restriction n), z) = Q(S(z\restriction n), z)\}.$$

Theorem 7.10.2 The space CM is complete in the metric δ. □

Proof Let $(P_i)_{i:\mathbb{N}}$ be a δ-Cauchy sequence of cpm's, i.e.

$$\forall \epsilon > 0, \exists\, N : \mathbb{N} \cdot \ \forall\, n, m > N \cdot \ \delta(P_n, P_m) < \epsilon.$$

This implies that for every $z \in \Sigma^\omega$

$$\sum_{k=0}^{\infty} \frac{1}{2^k} |P_n(S(z\restriction k), z) - P_m(S(z\restriction k), z)| < \epsilon.$$

Since $P_n(S(t), z) = 0$ for any $t \not\leq z$ we have

$$\sum_{k=0}^{\infty} \frac{1}{2^k} \sum_{t \in \Sigma^k} |P_n(S(t), z) - P_m(S(t), z)| < \epsilon,$$

that is for every $z \in \Sigma^\omega$ the sequence of P_n's given z is a sequence of probability measures which is a Cauchy sequence with respect to the metric d. By theorem 4.2.4 this converges to a probability measure, P given z, say. For all $z \in \Sigma^\omega$ and all cylinder sets $A \in \mathcal{F}$ we have $P_n(A, z) \to P(A, z)$. Hence for a fixed cylinderset A, $P(A, z)$ is a random variable. Also if $A_n \uparrow A$ then $P(A_n, z) \to P(A, z)$ and as a function of z, $P(A, z)$ is the limit of the random variables $P(A_n, z)$ and hence itself a random variable. Hence the class of sets for which P is a random variable contains the cylinder sets and is a monotonic class. Hence it contains $\mathcal{F}$. So P is a cpm and the limit of the sequence $(P_n)_{n:\mathbb{N}}$. Therefore CM is δ-complete. □

The following theorem establishes Lipschitz bounds on the operators of $PCSP$.

Lemma 7.10.3 Let P, Q be terms possibly involving the term variable Z and let F and G be the corresponding semantic functions, that is let $F \triangleq M(Z, P)\rho$ and $G \triangleq M(Z, Q)\rho$. Consider a semantic function H such that

1. H is constant with respect to $\rho(\!|Z|\!)$. Then $r(H) = 0$.
2. $H = M(Z, Z)\rho$. Then $r(H) = 1$.
3. $H = M(Z, a \to P)\rho$. Then $r(H) \leq 1/2\ r(F)$.
4. $H = M(Z, e : E \to P_e)\rho$. Then $r(H) \leq 1/2\ \max_{e \in E}(r(F_e))$ where $F_e \triangleq M(Z, P_e)\rho$.
5. $H = M(Z, P\ {}_p\sqcap\ Q)\rho$. Then $r(H) = p\ r(F) + (1-p)\ r(G)$.
6. $H = M(Z, P\ {}_S\ \Box\ Q)\rho$. Then $r(H) \leq \max(r(F), r(G))$.
7. $H = M(Z, P\ {}_B\|_C\ Q)\rho$. Then $r(H) < 2(r(F) + r(G))$.

□

Proof Let X, Y be cpm's. If H is constant then $\delta(H\ X, H\ Y) = 0$. If H is the identity function then $\delta(H\ X, H\ Y) = \delta(X, Y)$. For $H = M(Z, e : E \to P_e)\rho$ and $F_e \triangleq M(Z, P_e)\rho$ we have

$$
\begin{aligned}
&\delta(H\ X, H\ Y) \\
&\quad = \sup_{z \in \Sigma^\omega} \sum_{n=1}^{\infty} \frac{1}{2^n} \\
&\qquad (|\sum_{e \in E} I_{S(e)}(z)(F_e X\ (prefix_e^{-1} S(z \restriction n), z/1) - F_e Y\ (prefix_e^{-1} S(z \restriction n), z/1)) \\
&\qquad + \sum_{e \notin E} I_{S(e)}(z)((\!|STOP|\!)\ (S(z \restriction n), z) - (\!|STOP|\!)\ (S(z \restriction n), z))|)
\end{aligned}
$$

$$\begin{aligned}
&\leq \sup_{z/1\in\Sigma^\omega} \sum_{n=1}^{\infty} \frac{1}{2^n} \\
&\qquad \sum_{e\in E} I_{S(e)}(z)|F_e X\,(prefix_e^{-1} S(z\restriction n), z/1) - F_e Y\,(prefix_e^{-1} S(z\restriction n), z/1)| \\
&\leq \frac{1}{2} \max_{e\in E} \delta(F_e X, F_e Y).
\end{aligned}$$

So $r(H) \leq 1/2\ \max_{e\in E}(r(F_e))$. The Lipschitz condition for prefixing arises as a special case of this rule. For an expression with probabilistic choice the metric δ is

$$\begin{aligned}
&\delta(H\ X, H\ Y) \\
&= \sup_{z\in\Sigma^\omega} \sum_{n=1}^{\infty} \frac{1}{2^n} \mid p\,(F\ X\ (S(z\restriction n), z) - F\ Y\ (S(z\restriction n), z)) \\
&\qquad\qquad +(1-p)\,(G\ X\ (S(z\restriction n), z) - G\ Y\ (S(z\restriction n), z))| \\
&\leq p\ \delta(F\ X, F\ Y) + (1-p)\ \delta(G\ X, G\ Y).
\end{aligned}$$

For general choice, i.e. if $H = M(Z, P_S \,\square\, Q)\rho$, we have

$$\begin{aligned}
&\delta(H\ X, H\ Y) \\
&= \sup_{z\in\Sigma^\omega} \sum_{n=1}^{\infty} \frac{1}{2^n} |I_S(z)(F\ X\ (S(z\restriction n), z) - F\ Y\ S(z\restriction n), z)) \\
&\qquad\qquad +I_{S^c}(z)(G\ X\ (S(z\restriction n), z) - G\ Y\ (S(z\restriction n), z))| \\
&\leq \max(\delta(F\ X,\ F\ Y), \delta(G\ X, G\ Y)).
\end{aligned}$$

So $r(H) = \max(r(F), r(G))$.

To determine the Lipschitz condition of parallel composition recall that for τ-free z, i.e. if $z \in \Sigma^\omega$, then

$$(\!|P\ {}_B\|_C\ Q|\!)(S(z\restriction n), z) \;=\; (\!|P|\!)(S((z\restriction n)\restriction B), z\restriction B)\ (\!|Q|\!)(S((z\restriction n)\restriction C), z\restriction C).$$

Therefore if $H = M(Z, P\ {}_B\|_C\ Q)\rho$ then

$$\begin{aligned}
&\delta(H\ X, H\ Y) \\
&= \sup_{z\in\Sigma^\omega} \sum_{n=1}^{\infty} \frac{1}{2^n} (|\ F\ X\ (S((z\restriction n)\restriction B), z\restriction B)\ G\ X\ (S((z\restriction n)\restriction C), z\restriction C)|) \\
&\qquad\qquad -F\ Y\ (S((z\restriction n)\restriction B), z\restriction B)\ G\ Y\ (S((z\restriction n)\restriction C), z\restriction C) \\
&< \sup_{z\in\Sigma^\omega} \sum_{n=1}^{\infty} \frac{1}{2^n} (|F\ X\ (S((z\restriction n)\restriction B), z\restriction B) - F\ Y\ (S((z\restriction n)\restriction B), z\restriction B)| \\
&\qquad\qquad +|\ G\ X\ (S((z\restriction n)\restriction C), z\restriction C) - G\ Y\ (S((z\restriction n)\restriction C), z\restriction C)|)
\end{aligned}$$

where the last step follows from the inequality 4.1. Suppose that the first k elements of a trace z are in B and the next j in $C - B$. Then for $n < k$, $(z\restriction n)\restriction B = (z\restriction B)\restriction n$.

For $k \leq n < k+j$, $(z \restriction n) \restriction B = (z \restriction B) \restriction k$. Also $\sum_{n=k}^{k+j} 2^{-n} \leq 2.2^{-k}$. Thereafter $(z \restriction n) \restriction B \leq (z \restriction B) \restriction n-j)$ and for any subsequence of z consisting of i actions in $C - B$ we have $2^{-n} < 2.2^{-(n-j)}$. Hence

$$\begin{aligned}
&\delta(H\ X, H\ \ Y) \\
&\leq \sup_{z \in B^\omega} \sum_{n=1}^{\infty} 2\frac{1}{2^n} |F\ X\ (S(z \restriction n), z) - F\ \ Y\ (S(z \restriction n), z)| \\
&\quad + \sup_{z \in C^\omega} \sum_{n=1}^{\infty} 2\frac{1}{2^n} |G\ X\ (S(z \restriction n), z) - G\ \ Y\ (S(z \restriction n), z)| \\
&\leq \delta(F\ X,\ F\ Y) + \delta(G\ X,\ G\ Y).
\end{aligned}$$

So $r(H) < 2(r(F) + r(G))$. □

As in chapter 4, the Lipschitz bounds for δ mean that even unguarded recursion is sometimes well defined. However, we still need δ' to show that guarded recursion is always well defined.

Definition 7.10.4 Let P be a *PCSP*-term possibly involving a free variable Z. We say that P is *constructive* if $M(Z, P)\rho$ is a contraction map w.r.t. δ', and *non-destructive* if $M(Z, P)\rho$ is non-expanding w.r.t. δ'. □

This means that

$$\begin{aligned}
&P \text{ is constructive} \\
&\Leftrightarrow \delta'((\!|P|\!)\rho[X/Z], (\!|P|\!)\rho[Y/Z]) < \delta'(X, Y) \\
&\Leftrightarrow (\forall z \in \Sigma^\omega \cdot X\ (S(z \restriction n), z) = Y\ (S(z \restriction n), z) \\
&\qquad \Rightarrow \forall z \in \Sigma^\omega \cdot (\!|P|\!)\rho[X/Z](S(z \restriction n+1), z) = (\!|P|\!)\rho[Y/Z](S(z \restriction n+1), z)).
\end{aligned}$$

Similarly for non-destructive terms.

Lemma 7.10.5

1. *STOP* is constructive.
2. X is non-destructive.
3. $a \rightarrow P$ is constructive if P is non-destructive.
4. $e : E \rightarrow P_e$ is constructive if every P_e is non-destructive.
5. $P\ {}_p\sqcap\ Q$, $P\ {}_S\Box\ Q$, $P \parallel Q$ and $P\ {}_B\|_C\ Q$ are constructive if P and Q are constructive.

□

Proof Let X, Y be two cpm's and suppose that $\forall z \in \Sigma^\omega \cdot X(S(z \restriction n), z) = Y(S(z \restriction n), z)$. Clauses 1 and 2 follow directly. For clause 4 (which implies clause 3) note that

$$\begin{aligned}
&(\!|e : E \to P_e|\!)\rho[X/Z]\,(S(z \restriction n+1), z) \\
&= \begin{cases} (\!|P_{z_0}|\!)\rho[X/Z]\,(S((z/1) \restriction n), z) & \text{if } z_0 \in E \\ 0 & \text{otherwise} \end{cases} \\
&= \begin{cases} (\!|P_{z_0}|\!)\rho[Y/Z]\,(S((z/1) \restriction n), z) & \text{if } z_0 \in E \\ 0 & \text{otherwise} \end{cases} \\
&= (\!|e : E \to P_e|\!)\rho[Y/Z]\,(S(z \restriction n+1), z).
\end{aligned}$$

The proof that probabilistic choice is non-destructive is the same as in chapter 4 if we substitute $(S((z/1) \restriction n), z)$ for the probability measures used in the argument there. The proof that general choice is non-destructive follows if we substitute the appropriate indicator functions for the probability of choice.

For alphabetised parallel composition we get

$$\begin{aligned}
&(\!|P \,{}_B\|_C\, Q|\!)\rho[X/Z](S(z \restriction n), z) \\
&= (\!|P|\!)\rho[X/Z](S((z \restriction n) \restriction B), z \restriction B)\ (\!|Q|\!)\rho[X/Z](S((z \restriction n) \restriction C), z \restriction C) \\
&= (\!|P|\!)\rho[Y/Z](S((z \restriction n) \restriction B), z \restriction B)\ (\!|Q|\!)\rho[Y/Z](S((z \restriction n) \restriction C), z \restriction C) \\
&= (\!|P \,{}_B\|_C\, Q|\!)\rho[Y/Z](S(z \restriction n), z).
\end{aligned}$$

This completes the proof. □

We combine the last two lemmas to characterise a class of recursive expressions which are well-defined.

Theorem 7.10.6 Suppose that P is a $PCSP$ expression possibly containing the free variable X. If $r(M(X, P)\rho) < 1$ or if P is constructive with respect to X then the semantics

$$(\!|\mu X \bullet P|\!)\rho$$

is well defined for all bindings ρ. □

For well-defined $\mu X \bullet P$ the same laws apply for $PCSP$ as for $PCSP_0$.

Lemma 7.10.7

L6 $\mu X \bullet P \equiv P[\mu X \bullet P/X]$.

L7 If Y is not free in P then $\mu X \bullet P \equiv \mu Y \bullet P$.

L8 If $M(X,P)\rho$ is a δ-contraction map then $\mu X \bullet (P \,{}_{\mathbf{p}}\!\sqcap\, X) \equiv \mu X \bullet P$.

□

We treat mutual recursion in the same way as in chapter 4, substituting cpm's for probability measures as appropriate. Furthermore, the similarity between the metric d' on PM and δ' on CM means that recursion induction on cpm's can be treated in the same way as recursion induction on probability measures. So we will be able to use the following two rules: Suppose that R is a satisfiable and continuous predicate and that the $PCSP$-term P is constructive for the variable X. Then

Rule 7.10.8

$$\frac{\forall\, Y : PM \cdot R(Y) \Rightarrow R(\llparenthesis P \rrparenthesis \rho[Y/X])}{R(\llparenthesis \mu X \bullet P \rrparenthesis \rho)}$$

□

If $\boldsymbol{P}$ is a vector of mutually recursive processes which is constructive for the vector of variables $\boldsymbol{X}$ then to establish that a vector of predicates $\boldsymbol{R}$ correctly describes the fixed point of $M(\boldsymbol{X},\boldsymbol{P})$ it is sufficient to show that each R_i is continuous and satisfiable and that $\boldsymbol{R}$ is preserved by $M(\boldsymbol{X},\boldsymbol{P})$.

Rule 7.10.9

$$\frac{(\forall\, i \cdot R_i(Y_i)) \Rightarrow \forall\, j \cdot R_j(\llparenthesis P_j \rrparenthesis \rho[\boldsymbol{Y}/\boldsymbol{X}])}{\boldsymbol{R}(\llparenthesis \mu \boldsymbol{X} \bullet \boldsymbol{P} \rrparenthesis \rho)}$$

□

7.11 Two Common Properties of *PCSP*-processes

We can now prove that all $PCSP$-processes have the following two properties.

Lemma 7.11.1

1. If P is a $PCSP$-term then $\forall\, y \in \Omega$

$$\llparenthesis P \rrparenthesis (\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}, y) \;=\; 1.$$

2. If $t \in \Sigma_\tau^*$ and $y > t$ then $\llparenthesis P \rrparenthesis (S(t), y) = \llparenthesis P \rrparenthesis (S(t), t\langle\tau\rangle^\omega)$.

□

Proof We prove both properties by structural induction. Consider property 1. It is obviously true of $STOP$. For the every operator, suppose that property 1 is true of the arguments. Then for external choice

$$\begin{aligned}
&(\!|e : E \to P_e|\!)(\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}, y) \\
&\quad \widehat{=} \sum_{e\in E} I_{S(e)}(y/1)\,(\!|P_e|\!)(prefix_e^{-1}\{y/1\} \cup \bigcup_n \{(y/1\restriction n)\langle\tau\rangle^\omega\}, y/1) \\
&\qquad + \sum_{e\notin E} I_{S(e)}(y)\,(\!|STOP|\!)(\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}, y) \\
&\quad = 1.
\end{aligned}$$

The proofs for probabilistic and general choice follow immediately from the inductive hypothesis. For parallel composition we have

$$cpar_{B,C,y}^{-1}(\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}) \;=\; B^\omega \times C^\omega.$$

Hence

$$\begin{aligned}
(\!|P\ {}_B\|_C\ Q|\!)(\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}, y) &= (\!|P|\!)(B^\omega, y\restriction B)\,(\!|Q|\!)(C^\omega, y\restriction C) \\
&= 1.
\end{aligned}$$

The proof for relabelling follows because the relabelling function f is 1-1 and maps τ to itself. Hence

$$f^{-1}(\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}) \;=\; \{f^{-1}y\} \cup \bigcup_n \{((f^{-1}y)\restriction n)\langle\tau\rangle^\omega\}.$$

A well defined recursive process $\mu X \bullet P$ is the limit of a sequence of iterates, $(F^n(\!|STOP|\!))_{n:\mathbb{N}}$ say, each of which satisfies property 1. Since $\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}$ is a closed set, we deduce from theorem 2.1.6 that

$$\begin{aligned}
&(\!|\mu X \bullet P|\!)(\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}, y) \\
&\quad \geq \limsup F^n(\!|STOP|\!)(\{y\} \cup \bigcup_n \{(y\restriction n)\langle\tau\rangle^\omega\}, y) \\
&\quad = 1.
\end{aligned}$$

Now consider property 2. It is obviously true of $STOP$. It is true of all processes if $t = \langle\rangle$. For non-empty t, suppose that it holds of the arguments of each $PCSP$-operator. If $t_0 \in E$ then

$$\begin{aligned}
(\!|e : E \to P_e|\!)(S(t), y) &= P_e(S(t/1), y/1) \\
&= (\!|P_e|\!)(S(t/1), (t/1)\langle\tau\rangle^\omega) \\
&= (\!|e : E \to P_e|\!)(S(t), t\langle\tau\rangle^\omega).
\end{aligned}$$

If $t_0 \notin E$ then the process behaves as $STOP$ and property 2 is also true. For probabilistic choice the proof follows directly from the inductive hypothesis. For general choice we have

$$(\!|P\ {}_S\,\Box\, Q|\!)(S(t), y) \;=\; I_S(y)\,(\!|P|\!)(S(t), y) + I_{S^c}(y)\,(\!|Q|\!)(S(t), y).$$

If $S \cap S(t) = \emptyset$ this becomes

$$\begin{aligned} (\!|P \;_S \Box\; Q|\!)\,(S(t), y) &= (\!|Q|\!)\,(S(t), y) \\ &= (\!|Q|\!)\,(S(t), t\langle\tau\rangle^\omega) \\ &= (\!|P \;_S \Box\; Q|\!)\,(S(t), t\langle\tau\rangle^\omega). \end{aligned}$$

Similarly if $S^c \cap S(t) = \emptyset$. If $S(t)$ intersects both S and S^c then by definition

$$y \in S \cap S(t) \wedge z \in S^c \cap S(t) \;\Rightarrow\; (\!|P|\!)\,(S(t), y) = (\!|Q|\!)\,(S(t), z).$$

Hence the above steps also apply to this case. For parallel composition we have

$$\begin{aligned} (\!|P \;_B\|_C\; Q|\!)\,(S(t), y) &= (\!|P|\!)\,(S(t \restriction B), y \restriction B)\;(\!|Q|\!)\,(S(t \restriction C), y \restriction C) \\ &= (\!|P|\!)\,(S(t \restriction B), (t \restriction B)\langle\tau\rangle^\omega)(\!|Q|\!)\,(S(t \restriction C), (t \restriction C)\langle\tau\rangle^\omega) \\ &= (\!|P \;_B\|_C\; Q|\!)\,(S(t), t\langle\tau\rangle^\omega). \end{aligned}$$

The case of relabelling is easily checked. Finally, recursive processes satisfy property 2, because the probability of $S(t)$ is preserved in the limit. This concludes the proof. □

Chapter 8

Proof Rules

The description of an algorithm in *CSP* process algebra is a specification at an intermediate level of abstraction. At a higher level of abstraction, the properties which the algorithm is designed to achieve provide predicates upon process behaviours. If it can be shown that such a predicate holds of every possible behaviour of the algorithm we say that the algorithm satisfies the predicate. At a lower level of abstraction lies a (deterministic) implementation in, say, occam.

In the traces model of *CSP*, a behaviour is just a finite trace. In the failures-divergences model it is represented as a failure, that is as a trace combined with refusal sets. In the timed model, a behaviour is a timed failure. In all these models the semantics of a process is the set of all possible behaviours of that process. Thus to characterise when a process P with semantics $\mathcal{T}[\![P]\!]$ satisfies a specification expressed as a predicate R with free variable u, it is sufficient to set

$$P \textbf{ sat } R \;\widehat{=}\; (u \in \mathcal{T}[\![p]\!] \Rightarrow R(u)).$$

In the probabilistic model a process behaviour is an infinite trace. The semantics of a process is not the set of all possible behaviours, but a cpm which assigns a probability to every behaviour. One way of defining when a *PCSP*-process satisfies a specification would be to derive the set of possible behaviours from the cpm and use it in a definition of the above form. However, we can give a more direct definition. If a predicate describes all the possible behaviours of a process, then the probability of any behaviour for which the predicate is false must be zero. So we say

$$P \textbf{ sat } R \;\widehat{=}\; \forall z \in \Omega \cdot (\!|P|\!)(R, z) = 1$$

where we follow the convention of writing just R for the set $\{u \mid R(u)\}$. Note that since every process behaves like *STOP* when offered $\langle\tau\rangle^\omega$ by the environment, i.e. $\forall P \in CM \cdot P(\{\langle\tau\rangle^\omega\}, \langle\tau\rangle^\omega) = 1$, it follows that P **sat** $R \Rightarrow R(\langle\tau\rangle^\omega) = true$. So

typically R is a predicate of the form $(u_0 = \tau) \vee R'(u)$, that is a predicate which constrains what a process may do if it does anything at all. Such a constraint is called a *safety* property, as opposed to a *liveness* property which asserts that a process will do something.

If a safety property is violated, then at some finite point some undesired behaviour occurs which is irremediable. For instance, the statement that certain actions always happen in the same order constitutes a safety property because once the order has been violated, it cannot be restored by any later actions.

By contrast, a liveness property can be satisfied at some point in the future no matter what happens initially. Typical liveness properties are fairness, asymptotic behaviours, starvation freedom and termination. These observations motivate the following definitions which we adopt from Alpern and Schneider [AS85].

Definition 8.0.2 A predicate R upon infinite sequences with free variable u represents a *safety property* if

$$\forall u : \Omega \cdot \neg R(u) \Rightarrow \exists n : \mathbb{N} \cdot S(u \restriction n) \cap R = \emptyset.$$

□

Definition 8.0.3 A predicate R upon infinite sequences with free variable u represents a *liveness property* if

$$\forall t \in \Sigma^* \cdot S(t) \cap R \neq \emptyset.$$

□

Since these definitions are expressed in terms of infinite traces they cannot be used in a semantics which is based on finite traces, like the traces model or the failures-divergences model. In the traces model it is impossible to reason about liveness properties. In the failures-divergences model, liveness properties are expressed in terms of refusal sets. It would be interesting to investigate the differences between these alternative concepts of liveness, but we have not had time to address this issue.

8.1 Safety Properties

We now present an inference rule for each clause in the syntax of *PCSP*, expressing the properties of a process in terms of predicates with several components. For compound processes, the antecedent of the rule will consist of component specifications for the component processes.

The definition of **sat** gives rise to the usual logical rules:

$$\frac{}{P \textbf{ sat } \mathit{true}} \qquad \frac{P \textbf{ sat } R \quad P \textbf{ sat } T}{P \textbf{ sat } (R \wedge T)} \qquad \frac{P \textbf{ sat } R \quad R \Rightarrow T}{P \textbf{ sat } T}$$

The null specification is true of any process because $\{u \mid \mathit{true}\} = \Omega$ and

$$\forall z \in \Omega \cdot \ (\!|P|\!)(\Omega, z) = 1.$$

Each goal may be addressed separately because

$$\forall z \in \Omega \cdot (\!|P|\!)(R, z) = 1 \wedge (\!|P|\!)(T, z) = 1$$
$$\Rightarrow \ \forall z \in \Omega \cdot (\!|P|\!)(R \cap T, z) = 1.$$

We may weaken any specification already established because

$$\forall z \in \Omega \cdot (\!|P|\!)(R, z) = 1 \wedge R \subseteq T \ \Rightarrow \ \forall z \in \Omega \cdot (\!|P|\!)(T, z) = 1.$$

The process $STOP$ is unwilling to participate in any external activity. The first visible action performed by $a \rightarrow P$ must be a and the subsequent behaviour is that of P. So the inference rules for $STOP$ and $a \rightarrow P$ are

$$\frac{}{STOP \textbf{ sat } u = \langle\tau\rangle^{\omega}} \qquad \frac{P \textbf{ sat } R}{a \rightarrow P \textbf{ sat } (u_0 = \tau) \vee (u_0 = a \wedge R(u/1))}$$

These last two rules are special cases of the following:

$$\frac{\forall e \in E \cdot P_e \textbf{ sat } R_e}{c : E \rightarrow P_e \textbf{ sat } (u_0 = \tau) \vee (u_0 \in E \wedge R_{u_0}(u/1))}$$

To show that this is sound let $R(u) \hat{=} (u_0 = \tau) \vee (u_0 \in E \wedge R_e(u/1))$. Then $\forall z \in \Omega$

$$\begin{aligned}
&(\!|c : E \rightarrow P_e|\!)(R, z) \\
&= \sum_{e \in E} I_{S\langle e\rangle}(z)(\!|P_e|\!)(prefix_e^{-1} R, z/1) + \sum_{e \notin E} I_{S\langle e\rangle}(z)(\!|STOP|\!)(R, z) \\
&= \sum_{e \in E} I_{S\langle e\rangle}(z)(\!|P_e|\!)(R_e, z/1) + \sum_{e \notin E} I_{S\langle e\rangle}(z)(\!|STOP|\!)(R, z) \\
&= 1.
\end{aligned}$$

Any behaviour of the probabilistic choice $P \ {}_p\!\sqcap\ Q$ must arise from either P or Q. This gives rise to the inference rule

$$\frac{P \textbf{ sat } R \quad Q \textbf{ sat } T}{P \ {}_p\!\sqcap\ Q \textbf{ sat } (R \vee T)}$$

This is sound because for all $z \in \Omega$

$$\begin{aligned}(\!|P \;{}_p\!\cap\; Q|\!)\,(R \cup T, z) &= p\,(\!|P|\!)\,(R \cup T, z) + (1-p)\,(\!|Q|\!)\,(R \cup T, z)\\ &= p\,(\!|P|\!)\,(R, z) + (1-p)\,(\!|Q|\!)\,(T, z)\\ &= 1.\end{aligned}$$

A process offering external choice also behaves like one of its components. So the same inference rule applies as for probabilistic choice.

$$\frac{P \textbf{ sat } R \qquad Q \textbf{ sat } T}{P \;{}_S\!\Box\; Q \textbf{ sat } (R \vee T)}$$

This rule can be strengthened if we make the antecedents depend on the traces offered by the environment, in which case we have to abandon the sat-notation in the antecedent part of the rule.

$$\frac{\forall z \in S \cdot (\!|P|\!)\,(R, z) = 1 \qquad \forall z \in S^c \cdot (\!|Q|\!)\,(T, z) = 1}{P \;{}_S\!\Box\; Q \textbf{ sat } (R \vee T)}$$

The soundness of this rule (as of the weaker one) follows immediately from the definition of general choice:

$$\begin{aligned}(\!|P \;{}_S\!\Box\; Q|\!)\,(R \cup T, z) &= I_S(z)(\!|P|\!)\,(R \cup T, z) + I_{S^c}(\!|Q|\!)\,(R \cup T, z)\\ &= 1.\end{aligned}$$

In simple parallel composition $P \parallel Q$ processes P and Q must synchronise on every action. Thus the parallel system can do only what both of them are prepared to do:

$$\frac{P \textbf{ sat } R \qquad Q \textbf{ sat } T}{P \parallel Q \textbf{ sat } (R \wedge T)}$$

By definition,

$$(\!|P \parallel Q|\!)\,(R \cap T, z) = \int (\!|P|\!)\,((par^{-1}(R \cap T))_x, z)\,(\!|Q|\!)\,(dy, z).$$

We prove that

$$(\!|P|\!)\,(R, z) = 1 \wedge (\!|Q|\!)\,(T, z) = 1 \;\Rightarrow\; R \times T \subseteq par^{-1}(R \cap T).$$

We know that if a process attempts to perform a trace, u say, then the environment can force it to stop by offering a trace z which disagrees with u from some point onwards. Therefore if P **sat** R and $R(u)$ then R must also be true of all prefixes of u followed by $\langle\tau\rangle^\omega$:

$$P \text{ sat } R \;\Rightarrow\; (R(u) \;\Rightarrow\; \forall n \cdot R((u\restriction n)\langle\tau\rangle^\omega)).$$

Also $\forall v \cdot par(u,v) = u \vee \exists n \cdot par(u,v) = (u\restriction n)\langle\tau\rangle^\omega$. Thus $u \in R \Rightarrow par(u,v) \in R$. Applying the same argument to $v \in T$ we get

$$u \in R \wedge v \in T \;\Rightarrow\; par(u,v) \in R \cap T.$$

Therefore

$$\begin{aligned} &P \textbf{ sat } R \wedge Q \textbf{ sat } T \\ &\Rightarrow (\!|P \parallel Q|\!)(R \cap T, z) &&= ((\!|P|\!) \times (\!|Q|\!))(par^{-1}(R \cap T), z) \\ &&&\leq ((\!|P|\!) \times (\!|Q|\!))(R \times T, z) \\ &&&= (\!|P|\!)(R, z)\,(\!|Q|\!)(T, z) \\ &&&= 1. \end{aligned}$$

The generalised version of this inference rule is

$$\frac{\begin{array}{l} P \textbf{ sat } R \wedge (u \in B^\omega) \\ Q \textbf{ sat } T \wedge (u \in C^\omega) \end{array}}{P \;{}_B\|_C\; Q \textbf{ sat } R(u\restriction B) \wedge T(u\restriction C)}$$

The proof of soundness of this rule has to take account of B and C but apart from that follows along similar lines as the previous proof. Since R and T are 'prefix-closed' we have

$$\begin{aligned} &u \in R \wedge u \in B^\omega \wedge v \in T \wedge v \in C^\omega \\ &\Rightarrow\; cpar_{B,C,z}(u,v) \in \{w \mid R(w\restriction B) \wedge T(w\restriction C)\}. \end{aligned}$$

So

$$(R \cap B^\omega) \times (T \cap C^\omega) \;\subseteq\; cpar^{-1}_{B,C,z}\{w \mid R(w\restriction B) \wedge T(w\restriction C)\}.$$

Therefore

$$\begin{aligned} &P \textbf{ sat } (R \wedge u \in B^\omega) \wedge Q \textbf{ sat } (T \wedge v \in C^\omega) \\ &\Rightarrow\; (\!|P \;{}_B\|_C\; Q|\!)(\{u \mid R(u\restriction B) \wedge T(u\restriction C)\}, z) \\ &\qquad = ((\!|P|\!) \times (\!|Q|\!))(cpar^{-1}_{B,C,z}\{w \mid R(w\restriction B) \wedge T(w\restriction C)\}, z) \\ &\qquad \leq (\!|P|\!)(R, z\restriction B)\,(\!|Q|\!)(T, z\restriction C). \end{aligned}$$

The following proof rule enables us to show that a recursive process satisfies a predicate upon traces. let R be a safety property. Then

$$\frac{X \textbf{ sat } R \Rightarrow P \textbf{ sat } R}{\mu X \bullet P \textbf{ sat } R}$$

To show that this is sound we show that it is a special instance of the proof rule 7.10.9 (for predicates upon recursive processes). By definition, if $X \in CM$ then

$$X \textbf{ sat } R \quad \Leftrightarrow \quad \forall y \in \Omega \cdot X(R, y) = 1.$$

We know that R is 'prefix-closed' in the seuse that auy visible prefix of an element of R followed by $\langle\tau\rangle^\omega$ is itself an element of R. Let

$$T(X) \quad \hat{=} \quad \forall y \in \Omega \cdot X(R, y) = 1.$$

Then T is a satisfiable predicate upon processes, because $STOP$ **sat** R. If $T(X)$ is false, then X must assign positive probability to some behaviour which violates R. Since R is a safety property this must be apparent at some finite point, and T is false of every process in the open ball of processes which agree with X np to that point. So T is continuous and the above proof rule is an instance of rule 7.10.9.

8.2 Liveness Properties

The proof rules presented in the last section are most useful for safety properties. For liveness properties we have to assume that the environment does not block the progress of the system whose properties we are trying to prove, i.e. that the environment resolves every external choice on which the system depends, but accepts every internal (that is: probabilistic choice) made by the system. It turns out that any process combined with such an environment can be modelled simply as a probability measure, rather than as a cpm.

In this section we identify a subset of $PCSP$ which has a well-defined semantics both in CM and in PM. We show that the semantics $(\!|\ |\!)$ of a construct in this subset of the language is related to its semantics as given by $[\![\ ]\!]$ by a transformation function on traces. We then show that the assumption we make of the environment of a process to analyse its liveness properties results in a system that belongs to this subset of $PCSP$. So to analyse liveness properties we never have to consider cpm's, but only simple probability measures. That is, the same techniques which we used in chapter 5 to prove liveness properties of the rather limited class of $PCSP_0$ processes can also be used to analyse processes in general.

First note that every probability measure can be used to induce a conditional probability measure.

Lemma 8.2.1 If P is a probability measure in PM, then the function defined as

$$Q(A, y) \;\widehat{=}\; P\ cond_y^{-1} A$$

where

$$cond_y(x) \;\widehat{=}\; par(x, y)$$

is a cpm. □

Proof We know that par is measurable $(\mathcal{F} \times \mathcal{F})/\mathcal{F}$. Hence $cond_y$ is measurable $\mathcal{F}/\mathcal{F}$. So for fixed y, $P\ cond_y^{-1} A$ induces a probability measure. It remains to prove that for fixed $A \in \mathcal{F}$ the function $P\ cond_y^{-1} A$ is $\mathcal{F}$-measurable. Let $\mathcal{C}$ denote the class of sets such that for $C \in \mathcal{C}$ the functiou $P\ cond_y^{-1} C$ is $\mathcal{F}$-measurable. Suppose first that $C = S(t)$ where t is τ-free. Then

$$cond_y^{-1} S(t) \;=\; \begin{cases} S(t) & \text{if } y \in S(t) \\ \emptyset & \text{otherwise .} \end{cases}$$

Therefore $P\ cond_y^{-1} S(t) = I_{S(t)}(y)\ P\, S(t)$, which is a simple random variable. If t is not τ-free, the value of $P\ cond_y^{-1} S(t)$ can be computed as the difference of the probabilities of τ-free traces, i.e. as a difference of simple random variables. So in this case, too, $P\ cond_y^{-1} S(t)$ is a simple random variable. So $\mathcal{C}$ contains all the sets with fixed prefixes. It is easily shown that $\mathcal{C}$ is closed under finite unions and countable intersections. Therefore it is a monotone class and hence $\mathcal{C} = \mathcal{F}$. □

So for every probability measure we can construct a corresponding cpm. However, what we really need is to identify when a cpm has a corresponding probability measure.

Lemma 8.2.2 A $PCSP$-process $(\!|P|\!)$ has a corresponding probability measure $[\![P]\!]$ if and only if

$$\forall n : \mathbb{N} \cdot \sum_{t \in \Sigma^n} (\!|P|\!)(S(t), t\langle\tau\rangle^\omega) \;\leq\; 1.$$

□

Proof If t is a τ-free trace of length n then $cond_{t\langle\tau\rangle^\omega}^{-1} S(t) = S(t)$. Therefore if $(\!|P|\!)$ is a $PCSP$-process with a corresponding probability measure $[\![P]\!]$ then

$$\begin{aligned} [\![P]\!] S(t) \;&=\; [\![P]\!]\ cond_{t\langle\tau\rangle^\omega}^{-1} S(t) \\ &=\; (\!|P|\!)\,(S(t), t\langle\tau\rangle^\omega). \end{aligned}$$

This means that

$$\textstyle\sum_{t \in \Sigma^n} (\!|P|\!)(S(t), t\langle\tau\rangle^\omega) \;\leq\; 1$$

or else $[\![P]\!]$ would not be a probability measure. □

The next lemma provides a rule by which the existence of a probability measure corresponding to a cpm can be checked syntactically rather than by recourse to the semantics.

Lemma 8.2.3 If P is a $PCSP$-term containing only $STOP$, $\rightarrow$, ${}_p\sqcap$, $\|$ and possibly variables bound by recursion, then the semantics of P in $PCSP$ and $PCSP_0$ are related by

$$(\!|P|\!)(A,y) \;=\; [\![P]\!]\, cond_y^{-1} A.$$

□

Proof We use structural induction. To deal with $STOP$, note that $\langle\tau\rangle^\omega \in A \Leftrightarrow \langle\tau\rangle^\omega \in cond_y^{-1}A$. Therefore

$$\begin{aligned}(\!|STOP|\!)(A,y) &= I_A(\langle\tau\rangle^\omega)\\ &= I_{cond_y^{-1}A}(\langle\tau\rangle^\omega)\\ &= [\![STOP]\!]\, cond_y^{-1}A.\end{aligned}$$

To show that the equality is preserved by all the other operators, suppose that $(\!|P|\!)(A,y) = [\![P]\!]\, cond_y^{-1}A$. Note that

$$prefix_a\,;\, cond_y \;=\; \begin{cases} cond_{y/1}\,;\, prefix_a & \text{if } y_0 = a\\ cond_{\langle\tau\rangle^\omega} & \text{otherwise}.\end{cases}$$

So

$$\begin{aligned}(\!|a \rightarrow P|\!)(A,y) &= I_{S(a)}(y)\,(\!|P|\!)(prefix_a^{-1}A, y/1) + I_{S(a)^c}(y)\,(\!|STOP|\!)(A,y)\\ &\quad \text{by definition}\\ &= I_{S(a)}(y)\,(\!|P|\!)(prefix_a^{-1}A, y/1) + I_{S(a)^c}(y)\,(\!|P|\!)(A,\langle\tau\rangle^\omega)\\ &= \begin{cases} [\![P]\!]\, cond_{y/1}^{-1}(prefix_a^{-1}A) & \text{if } y_0 = a\\ [\![P]\!]\, cond_{\langle\tau\rangle^\omega}^{-1}A & \text{otherwise}\end{cases}\\ &= [\![P]\!]\, prefix_a^{-1}(cond_y^{-1}A)\\ &= [\![a \rightarrow P]\!](cond_y^{-1}A).\end{aligned}$$

For probabilistic choice suppose that P and Q satisfy the hypothesis. Then

$$\begin{aligned}(\!|P\ {}_p\sqcap\ Q|\!)(A,y) &= p\,(\!|P|\!)(A,y) + (1-p)(\!|Q|\!)(A,y)\\ &= [\![P]\!]\, cond_y^{-1}A + (1-p)[\![Q]\!]\, cond_y^{-1}A\\ &= [\![P\ {}_p\sqcap\ Q]\!]\, cond_y^{-1}A.\end{aligned}$$

For parallel composition note that $par\,;\,cond_y = (cond_y, cond_y)\,;\,par$. Therefore

$$\begin{aligned}
(\!|P \parallel Q|\!)\,(A,y) &= \int (\!|P|\!)\,((par^{-1}A)_*, y)\;(\!|Q|\!)(du, y) \\
&= \int [\![P]\!]\,((cond_y^{-1}, id)(par^{-1}A))_{cond_y(w)}\;[\![Q]\!]\,dw \\
&\quad \text{change of variable} \\
&= \int [\![P]\!]\,((cond_y^{-1}, cond_y^{-1})(par^{-1}A))_w\;[\![Q]\!]\,dw \\
&= \int [\![P]\!]\,(par^{-1}(cond_y^{-1}A))_w\;[\![Q]\!]\,dw \\
&= [\![P \parallel Q]\!]\;cond_y^{-1}A.
\end{aligned}$$

If in addition to the above operators a term P contains a free variable X such that the recursion $\mu X \bullet P$ is well-defined, then the Banach Fixed Point Theorem assures us that

$$\begin{aligned}
(\!|\mu X \bullet P|\!)\rho\,(A,y) &= \lim_{n\to\infty} F^n(\!|STOP|\!)\rho\,(A,y) \\
&\quad \text{where } F = \lambda\, Y \cdot (\!|P|\!)\rho[Y/X] \\
&= \lim_{n\to\infty} G^n[\![STOP]\!]cond_y^{-1}A \\
&\quad \text{where } G = \lambda\, Y \cdot [\![P]\!]\rho[Y/X] \\
&= [\![\mu X \bullet P]\!]\rho\;cond_y^{-1}A.
\end{aligned}$$

This completes the proof. □

8.3 A Self-stabilising Tokenring

This self-stabilising algorithm is due to [Herm90]. Its purpose is to pass a token around a cyclically arranged group of processes. The process in possession of the token can execute some task without interference from any other process. For our purposes the nature of the task is immaterial. Each process is in one of two states; it alternately reads the state of its left-hand neighbour and passes its own state to its right-hand neighbour. Every process which is in the same state as its left-hand neighbour is said to have a token. A process which doesn't have the token keeps its state. A process which has a token changes state with probability 1/2. This causes the token to pass to the next process. The total number of processes must be odd, so that under normal conditions all neighbouring processes bar one pair are in different states.

We will prove this algorithm to be self-stabilising in the sense that whatever their initial states, the processes eventually reach a state where exactly one token exists (i.e. spurious tokens disappear) and which is live in the sense that each process is guaranteed to receive the token infinitely often.

$$T(i,S) \;=\; i.S(i) \rightarrow (T(i \oplus 1, S[0/S(i)]) \,{}_{1/2}\sqcap\; T(i \oplus 1, S[1/S(i)])) \lhd S(i \ominus 1) = S(i) \rhd T(i \oplus 1, S))$$

where $S[b/S(i)]$ denotes the state S with its i^{th} element overwritten with the value b. To prove that $TS \equiv T$ we use recursion induction. Given a vector of processes $\boldsymbol{Y}$ define

$$(\boldsymbol{R}(\boldsymbol{Y}))(i,S) \;\hat{=}\; ((\boldsymbol{Y})_{(i,S)} = (\!|\|_{B_j}\, j.P(i,S)|\!|)$$

where

$$j.P(i,S) \;=\; \begin{cases} j.P_{S(j)} & \text{if } i \neq j \\ j!m \rightarrow (j.P_{0\ 1/2}\sqcap\ j.P_1 \lhd S(j \ominus 1) = S(j) \rhd j.P_{S(j)}) & \text{if } j = i. \end{cases}$$

The predicate $\boldsymbol{R}$ is continuous and satisfiable. Let $\boldsymbol{X}$ be a vector of term variables and let F be the function which corresponds to one unfolding of the recursion.

$$(F\,\boldsymbol{X})_{i,S} \;\hat{=}\; i.S(i) \rightarrow ((\boldsymbol{X})_{i\oplus 1, S[0/S(i)]}\,{}_{1/2}\sqcap\,(\boldsymbol{X})_{i\oplus 1, S[1/S(i)]} \lhd S(i \ominus 1) = S(i) \rhd (\boldsymbol{X})_{i\oplus 1,S}).$$

Since F is constructive the mapping $M(\boldsymbol{X}, F)\rho$ has a unique fixed point. We can therefore use rule 7.10.9 to show that R holds of the fixed point of $M(\boldsymbol{X}, F)\rho$. Suppose that

$$\forall\, i, S \cdot (\boldsymbol{Y})_{i,S} = (\!|\|_{B_j}\, j.P(i,S)|\!|)$$

where $j.P(i,S)$ as defined above. We need to show that this implies

$$\forall\, i, S \cdot ((\!|F|\!)\rho[\boldsymbol{Y}/\boldsymbol{X}])_{i,S} = (\!|\|_{B_j}\, j.P(i,S)|\!|).$$

This is true if we can argue syntactically that

$$(\forall\, i, S \cdot X_{i,S} \;\equiv\; \|_{B_j}\, j.P(i,S)) \;\Rightarrow\; (\forall\, i, S \cdot (F\,\boldsymbol{X})_{i,S} \;\equiv\; \|_{B_j}\, j.P(i,S)).$$

Substituting for $\boldsymbol{X}$ in F we get

$$(F\,\boldsymbol{X})_{i,S} \;\equiv\; i.S(i) \rightarrow ((\|_{B_j}\, j.P(i \oplus 1, S[0/S(i)]))\,{}_{1/2}\sqcap\,(\|_{B_j}\, j.P(i \oplus 1, S[1/S(i)])) \lhd S(i \ominus 1) = S(i) \rhd (\|_{B_j}\, j.P(i \oplus 1, S))).$$

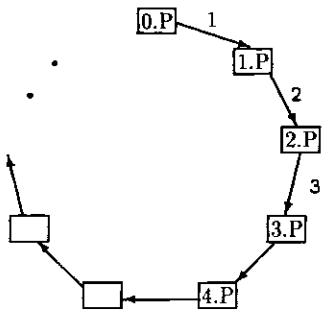

Figure 8.1: A tokeuring

Let the processes in the ring be numbered 0 to N, N even, and let the input channel to each process have the same number as the process (see Fig. 8.3). The tokenring consists of the $N+1$ processes operatiug in parallel:

$$T = \|_{B_i} i.P$$

where $0 \leq i \leq N$ and $B_i = \{j.b \mid j = i \oplus 1 \vee j = i \wedge b \in \{0,1\}\}$. The algorithm could start in any state, but for simplicity's sake we assume that every process starts out in state 0. The first process starts the cycle of communication.

$$\begin{aligned} 0.P &= 0!0 \rightarrow 0.P_0 \\ i.P &= i.P_0 \qquad i > 0. \end{aligned}$$

Every other process first asks for input from its left-hand neighbour aud outputs its own state to its right-hand neighbour. Then it decides whether or uot to change state. Let $\oplus$ and $\ominus$ denote addition and subtraction modulo $N+1$. For $m \in \{0,1\}$, $0 \leq j \leq N$ define

$$\begin{aligned} j.P_m = j?l \rightarrow j \oplus 1!m \rightarrow (&j.P_0 \ {}_{1/2}\sqcap \ j.P_1 \\ &\lhd l = m \rhd \\ &j.P_m). \end{aligned}$$

To analyse the tokenring, we first of all show that it can be represented in the following sequential form: Let $S \in \{0,1\}^{N+1}$ denote the $(N+1)$-tuple of the states of the processes. For $0 \leq i \leq N$ define

$$TS = T(0, \{0\}^{N+1})$$

Since $\|$ distributes through $\lhd\rhd$ we can bring every process not indexed by $j = i$ outside the if-statement and write

$$(F\ \boldsymbol{X})_{(i,S)} \equiv i.S(i) \to (\|_{B_j} j.Q)$$

where

$$j.Q = \begin{cases} j.P_{S(j)} & \text{if } j \neq i, j \neq i \oplus 1 \\ \quad i.P_0 \,{}_{1/2}\sqcap\ i.P_1 & \text{if } j = i \\ \lhd S(i \ominus 1) = S(i) \rhd & \\ \quad i.P_{S(i)} & \\ i \oplus 1.S(i \oplus 1) \to (\ i \oplus 1.P_0 \,{}_{1/2}\sqcap\ i \oplus 1.P_1 & \text{if } j = i \oplus 1 \\ \qquad \lhd S(i) = S(i \oplus 1) \rhd & \\ \qquad\quad i \oplus 1.P_{S(i \oplus 1)}) & \end{cases}$$

By law 9 for alphabetised parallel composition

$$\begin{aligned} i.S(i) \to (i.Q\ {}_{B_i}\|_{B_{i\oplus 1}}\ i \oplus 1.Q) &\cong (i!S(i) \to i.Q)\ {}_{B_i}\|_{B_{i\oplus 1}}\ (i?x \to i \oplus 1.Q) \\ &\equiv i.P(i,S)\ {}_{B_i}\|_{B_{i\oplus 1}}\ i \oplus 1.P(i,S). \end{aligned}$$

Hence

$$(F\ \boldsymbol{X})_{i,S} \equiv \|_{B_j} j.P(i,S).$$

So the antecedent of the proof rule for mutual recursion is true and we deduce that $\boldsymbol{Y}$ is the unique fixed point of $M(\boldsymbol{X}, F)\rho$. Therefore $\forall i, S$

$$T(i,S) \equiv \|_{B_j} j.P(i,S).$$

In particular

$$TS \equiv \|_{B_j} j.P(0,\{0\}^{N+1}).$$

Also

$$\begin{aligned} T &\triangleq \|_{B_j} j.P \\ &\equiv \|_{B_j} j.P(0,\{0\}^{N+1}). \end{aligned}$$

Therefore $T \equiv TS$. Since TS does not contain any external choice, it can be analysed as a probability measure rather than a cpm. Rather than repeating the proof of correctness given in [Herm90] we present an alternative proof which is slightly shorter. The difference is that [Herm90] start from first principles, whereas the proof given here exploits some general results about finite Markov chains.

Theorem 8.3.1 The tokenring is self-stabilising and live. □

Proof Clearly communication in the tokenring happens in rounds:

$$[\![T]\!]\{u : \Omega \mid chan(u_n) = n \bmod (N+1)\} \;=\; 1.$$

A communication on channel i affects only state $S(i)$. Therefore in one round of communication each part of the state may change at most once. Let $A(k, S)$ he the set of traces such that the states communicated in round k are S:

$$A(k.S) \;\widehat{=}\; \{u : \Omega \mid \forall 0 \le i \le N \cdot msg(u_{k(N+1)+i}) = S(i) \\ \wedge\; chan(u_n) = n \bmod (N+1)\}.$$

Let $t(S)$ be the number of tokens in the ring. If the total number of processes is odd, then at least one process must have a token and the total number of tokens is always odd, because new tokens cau only be generated two at a time.

The probability that the state of the tokenring is S' in round $k+1$ given that it was S in round k is

$$[\![T]\!](A(k+1), S') \mid A(k, S)) \\ = \begin{cases} \frac{1}{2}^i & \text{if } t(S) = i \wedge \forall j \cdot S(j \ominus 1) = S(j) \Rightarrow S'(j) = S(j) \\ 0 & \text{otherwise .} \end{cases}$$

Since there are only finitely many states, and the transition probability from one to the next does not depend on any previous states the sets $A(k, S)$ form a finite Markov chain. From a one-token state, only two transitions are possible. Both are again one-token states. So the set of states in which exactly one process has the token is a *closed* set in the seuse that the transition probabilities from any element of this set to any element outside this set are all zero. A state has more than one token if there exist j, k such that (w.o.l.o.g) $j < k$ and $S(j) = S(j \ominus 1)$ and $S(k) = S(k \ominus 1)$. Suppose that there is no token between j and k, i.e. $\forall l \cdot j < l < k \Rightarrow S(l) \neq S(l \ominus 1)$. From S, the ring can progress to the state $S' = S[/S(k)/S(k)]$ with non-zero probability. If $j = k-1$, i.e. if the tokens are adjacent, then the change from $S(k)$ to $/S(k)$ makes them disappear. If $k - j > 1$, i.e. if the tokens are more than 1 apart, then S' has a token at $j+1$ and k, that is the distance between the tokens has decreased by one. It follows that any state S with non-adjacent tokens j, k such that $k - j > 1$ has a non-zero probability of a transition in $k - j + 1$ steps to a state with two adjacent tokens at k and $k-1$, and hence a non-zero, $k - j$ step transition probability to a state with fewer tokens. Therefore all states with more than one token are *transient* in the sense that the probability of eventual return to this state is strictly less than one. In a finite Markov chain the probability of staying forever in a set of transient states is zero [Fel57]. So the tokenring will eventually end up in a state where exactly one process has the token, and from that point onwards the only other states it can visit are those where exactly one process has the token. Thus the tokenring is guaranteed to stabilise. (The result about transient states

means that the invariant proved by [Herm90], namely that the algorithm never increases the number of tokens, only needs to hold in the special case where the number of tokens is one.)

The closed set by itself represents a finite irreducible Markov chain, in which all states are persistent [Fel57]. i.e. all states are visited infinitely often. So the tokenring is live in the sense that every process is guaranteed to receive the token infinitely often. □

Chapter 9

Randomised Consensus

To illustrate the application of *PCSP* we give a formal specification and proof of correctness of a cousensus protocol. The specification is given at two levels of abstraction. At the top level the properties of a consensus protocol are defined by predicates upon traces. At a lower level a randomised algorithm which satisfies these properties is presented in the notation of *PCSP*. This algorithm is a variation of an algorithm which was developed by Aspnes and Herlihy [AH90]. Our version has the same safety properties, but slightly different liveness properties: whereas the algorithm by Aspnes and Herlihy is guaranteed to terminate under all circnmstances, the one used here terminates with probability 1 if the scheduling is independent of the state of the processes involved in the protocol. As a result we are able to reduce the expected number of steps to termination from Aspnes aud Herlihys's $\mathcal{O}(2^n)$ to $\mathcal{O}(n^2)$.

A *consensus protocol* is a procedure whereby N communicating processes which start out with conflicting preferences all come to agree on the same prefereuce. The final preference is called the *decision value*. A consensus protocol must be

1. *consistent*: no two processes choose different decision values,
2. *valid*: the decision value was some process's initial preference, and
3. *terminating*: every process that does not fail completes the procedure in finite expected time.

These properties represent the most abstract or high-level specification of the consensus protocol. An algorithm which the processes follow to reach a decision is correct if it satisfies the high level specification. To formalise this specification we use some shorthand for certain predicates: We write $a(u)$ to say that an action a occurs in a trace u:

$$a(u) \quad \widehat{=} \quad \exists\, n \in \mathbb{N} \cdot u_n = a$$

To say that a trace u contains actions a and b and a occurs before b we define

$$(a \textbf{ before } b)(u) \;\widehat{=}\; \exists n, m \in \mathbb{N} \cdot u_n = a \wedge u_m = b \wedge n < m$$

For convenience, we also define **after**:

$$(a \textbf{ after } b)(u) \;\widehat{=}\; (b \textbf{ before } a)(u)$$

Since it would be cumbersome to carry the dummy variable u all through the specification and proofs, we suppress it from now on. We write a before b before c as shorthand for a before $b \wedge b$ before c. Since the protocol consists of a collection of identical components, channels are indexed: we write $i.c.v$ to say that v is communicated on channel c belonging to the i^{th} component. Free variables for channel indices or message values can always be taken to be universally quantified.

9.1 Specification

The protocol consists of N processors which communicate by reading and writing N shared registers. Let I deuote the set of indices $\{i \mid 0 \leq i < N\}$. Fig 9.1 shows the

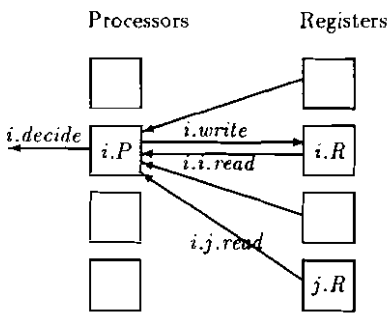

Figure 9.1: Channels connecting $i.P$

communication in which processor $i.P$, $i \in I$, can engage. It can read the values stored by register $j.R$, $j \in I$, via the channel $i.j.read$. It can write values to the i^{th} register via channel $i.write$. Finally it can communicate its decision about its final preference to the environment via channel $i.decide$. Each register stores a preference value and a round number. For simplicity's sake we assume that preference values are boolean and round numbers are natural numbers. The final decision consists of

just a preference value. Thus the set of actions which the process $i.P$ can perform is

$$\begin{aligned} B_i \;\hat{=}\; & \{i.j.read.(v,r) \mid j \in I \wedge v \in \mathbb{B} \wedge r \in \mathbb{N}\} \\ & \cup \{i.write.(v,r) \mid v \in \mathbb{B} \wedge r \in \mathbb{N}\} \\ & \cup \{i.decide.(v) \mid v \in \mathbb{B}\} \cup \{\tau\}. \end{aligned}$$

Correspondingly, the set of actions performed by the j^{th} register is

$$\begin{aligned} C_j \;\hat{=}\; & \{i.j.read.(v,r) \mid i \in I \wedge v \in \mathbb{B} \wedge r \in \mathbb{N}\} \\ & \cup \{j.write(v,r) \mid v \in \mathbb{B} \wedge r \in \mathbb{N}\} \cup \{\tau\}. \end{aligned}$$

The consensus protocol CP is the parallel composition

$$\begin{aligned} CP \;\hat{=}\; & (\|_{B_i}\, i.P) \;{}_B\|_C\; (\|_{C_j}\, j.R) \\ & \text{where } \mathrm{B} = \bigcup_i \mathrm{B_i},\ \mathrm{C} = \bigcup_j \mathrm{C_j}. \end{aligned}$$

We now give the formal definitions of its properties. Unless otherwise stated, the indices i, j, k range over I, preference values v over the booleans, and roundnumbers r over the natural numbers.

Safety

A decision appears in the traces as a *decide*-event. It is valid only if it was some processor's initial preference. The initial preference of a processor is the one which it writes to the register in round 1. Thus if a valid trace contains a *decide*-event with value v then it must also contain a first-round *write*-event with value v:

$$VS \;\hat{=}\; (i.decide.(v) \;\Rightarrow\; \exists j \cdot j.write.(v,1) \textbf{ before } i.decide.(v)).$$

Consistency requires that all processors make the same decision. So no consistent trace contains *decide*-events with different preference values:

$$CS \;\hat{=}\; \neg(i.decide.(1) \wedge j.decide.(0)).$$

Obviously validity and consistency are safety properties in the sense of definition 8.0.2.

Liveness

The protocol terminates if every process which is scheduled infinitely often must come to a decision. Thus the traces of a terminating protocol are described by

$$TS(u) \;\hat{=}\; \forall i \cdot (u \restriction B_i \text{ infinite } \Rightarrow \exists v \cdot i.decide.(v))$$

This is obviously a liveness property in the sense of definition 8.0.3.

The Algorithm

We now specify an algorithm which enables the processors to reach a decision. Each processor chooses au initial preference value for round 1 and writes it into its register. From round 1 until it can decide each processor alternately reads all the registers and, based on the values it has just read, writes a new preference value and ronndnumber. Let $\boldsymbol{v}$ and $\boldsymbol{r}$ be vectors of N preferred values and N roundnumbers respectively. Suppose that processor $i.P$ has just read the values $\boldsymbol{v}, \boldsymbol{r}$ from the registers. If according to these values it is a leader, that is one of the processors with the highest roundnumher, and all dissenting processors trail by at least two roundnumbers, it cau decide on a final preference. The condition for this case is expressed by

$$i.can\_decide(\boldsymbol{v}, \boldsymbol{r}) \;\widehat{=}\; r_i > 1 \wedge \forall j \cdot (r_j \leq r_i \wedge (v_j = v_i \vee r_i > r_j + 1)).$$

If a processor cannot decide it adopts if possible the preference of the leaders. If the leaders do not have a common preference, it sticks to its own preference, but randomly either advances to the next round or stays at the same round. Let $leaders\_agree(v, \boldsymbol{v}, \boldsymbol{r})$ denote the fact that based on the observed values $\boldsymbol{v}$ and $\boldsymbol{r}$ the leaders all prefer the same value v:

$$leaders\_agree(v, \boldsymbol{v}, \boldsymbol{r}) \;\widehat{=}\; \forall j \cdot (r_j = \max(\boldsymbol{r}) \Rightarrow v_j = v).$$

Then the processors are described by

$$\begin{aligned}
i.P &= i.write!(1,1) \rightarrow i.P(\emptyset, \emptyset, 0) \\
&\quad {}_p\sqcap\ i.write!(0,1) \rightarrow i.P(\emptyset, \emptyset, 0) \\
i.P(\boldsymbol{v}, \boldsymbol{r}, j) &= i.j.read?(v, r) \rightarrow i.P(\boldsymbol{v}', \boldsymbol{r}', j+1) \qquad j < N \\
&\quad \text{where } \boldsymbol{v}' = \boldsymbol{v} \cup \{j \mapsto v\},\ \boldsymbol{r}' = \boldsymbol{r} \cup \{j \mapsto r\} \\
i.P(\boldsymbol{v}, \boldsymbol{r}, N) &= i.decide!(v_i) \rightarrow IDLE \\
&\qquad \lhd i.can_decide(\boldsymbol{v}, \boldsymbol{r}) \rhd \\
&\quad (i.write!(v, r_i + 1) \rightarrow i.P(\emptyset, \emptyset, 0) \\
&\qquad \lhd \exists\, v \cdot leaders_agree(v, \boldsymbol{v}, \boldsymbol{r}) \rhd \\
&\quad (i.write!(v_i, r_i) \rightarrow i.P(\emptyset, \emptyset, 0)) \\
&\qquad {}_p\sqcap\ (i.write!(v_i, r_i + 1) \rightarrow i.P(\emptyset, \emptyset, 0))).
\end{aligned}$$

The process $IDLE$ can be any process which does not affect the state of the protocol, and need not be specified explicitely. A register starts in round 0. Afte· the first write it always produces the value that was last written to it.

$$j.R \;=\; j.R(0,0)$$

$$j.R(v,r) \;=\; j.write?(v',r') \to j.R(v',r') \;\Box\; \Box_{i\in I}\, i.j.read!(v,r) \to j.R(v,r).$$

9.2 Proof of Correctness

We will show that the protocol is correct in the sense that it satisfies the safety properties, i.e.

$$CP \;\textbf{sat}\; VS \wedge CS.$$

As explained in section 8.2, we can only reason about the liveness properties of a system if we assume that it lives in an environment which resolves every external choice on which the system depends, but accepts every internal choice of the system. In the case of the randomised consensus protocol the only choice to be resolved by the environment concerns the interleaving of the processors; given the opportunity to take a step, each processor will determine internally what this step should be. This means that we make an assumption about the probability distribution D which determines the interleaving and for which we have to show that

$$[\![CP \parallel D]\!]\; TS \;=\; 1.$$

To show that the safety properties are satisfied we first list the predicates PS_i which are satisfied by the processors and the predicates RS_j which are satisfied by the registers.

After the first round any value read from a register must be the last value that was written to it:

$$\begin{aligned} RS_j \;\triangleq\; & i.j.read(v,r) \Rightarrow \\ & r = 0 \vee \\ & j.write.(v,r) \textbf{ before } i.j.read(v,r) \wedge \\ & \neg \exists\, v', r' \cdot ((v' \neq v \vee r' \neq r) \wedge \\ & \quad j.write.(v,r) \textbf{ before } j.write.(v',r') \textbf{ before } i.j.read(v,r)). \end{aligned}$$

We write $i.Read(\boldsymbol{v},\boldsymbol{r})$ to say that processor $i.P$ has consecutively read all the registers and thus obtained the values $(\boldsymbol{v},\boldsymbol{r})$:

$$i.Read(\boldsymbol{v},\boldsymbol{r})(u) \;\triangleq\; \exists\, m \in \mathbb{N}, \forall j \cdot (u \restriction B_i)_{m+j} = i.j.read.(v_j, r_j).$$

We extend our notation and write $i.Read(\boldsymbol{v},\boldsymbol{r})$ **before** a, meaning that all the readings $i.j.read(v_j,r_j)$ were taken before the action a happened. Similarly for a **before** $i.Read(\boldsymbol{v},\boldsymbol{r})$, a **after** $i.Read(\boldsymbol{v},\boldsymbol{r})$ and $i.Read(\boldsymbol{v},\boldsymbol{r})$ **after** a.

A processor $i.P$ can decide only if it has read the registers and $i.can\_decide(\boldsymbol{v}, \boldsymbol{r})$ is true:

$$\begin{aligned} PS1_i \;\hat{=}\; & i.decide.(v) \Rightarrow \\ & \exists \boldsymbol{v}, \boldsymbol{r} \cdot i.decide.(v) \textbf{ after } i.Read(\boldsymbol{v}, \boldsymbol{r}) \\ & \wedge\, v_i = v \wedge i.can\_decide(\boldsymbol{v}, \boldsymbol{r}). \end{aligned}$$

Before writing a new value a processor must have read all the registers. It it switched preference it must have seen all the leaders disagree with its old preference. If it kept its own preference and advanced its roundnnmber, it cannot have seen the leaders agree on the opposite preference. If it kept its own preference and did not advance its ronndnumber, the leaders did not have a common preference.

$$\begin{aligned} PS2_i \;\hat{=}\; & i.write.(v, r+1) \Rightarrow \\ & \exists \boldsymbol{v}, \boldsymbol{r} \cdot i.write.(v, r+1) \textbf{ after } i.Read(\boldsymbol{v}, \boldsymbol{r}) \wedge \neg i.can\_decide(\boldsymbol{v}, \boldsymbol{r}) \\ & \wedge\, ((v_i \neq v \wedge r_i = r \wedge leaders\_agree(v, \boldsymbol{v}, \boldsymbol{r})) \\ & \vee\, (v_i = v \wedge r_i = r \wedge \neg leaders\_agree(1-v, \boldsymbol{v}, \boldsymbol{r})) \\ & \vee\, (v_i = v \wedge r_i = r+1 \wedge \neg \exists q \cdot leaders\_agree(q, \boldsymbol{v}, \boldsymbol{r}))). \end{aligned}$$

After a process has decided it cannot write any more values

$$PS3_i \;\hat{=}\; \neg(i.write.(v, r) \textbf{ after } i.decide.(w)).$$

Let $PS_i \mathrel{\hat{=}} PS1_i \wedge PS2_i \wedge PS3_i$. By the inference rule for parallel composition,

$$\frac{\begin{array}{l} \forall i \cdot i.P \textbf{ sat } PS_i \wedge (u \in B_i^{\omega}) \\ \forall j \cdot j.R \textbf{ sat } RS_j \wedge (u \in C_j^{\omega}) \end{array}}{(\|_{B_i} i.P) \parallel (\|_{C_j} j.R) \textbf{ sat } PS_i(u \restriction B_i) \wedge RS_j(u \restriction C_j)}$$

we know that the behaviour of the protocol restricted to the alphabet of a component must satisfy the same predicate as that component. The remainder of the proof of correctness is based on only one other proof rule, namely

$$\frac{\begin{array}{l} P \textbf{ sat } R \\ R \Rightarrow T \end{array}}{P \textbf{ sat } T.}$$

We first note that the predicates we are considering are such that if they hold of a trace restricted to the alphabet of a component they also hold of the unrestricted trace, i.e.

$$\begin{aligned} RS_j(u \restriction C_j) &\;\Leftrightarrow\; RS_j(u) \\ PS_i(u \restriction B_i) &\;\Leftrightarrow\; PS_i(u). \end{aligned}$$

Thus all we need to show is that the simple conjunction of the predicates PS_i and RS_j implies validity and consistency.

Validity

Recall that a consensus protocol is valid only if the decision value was the iuitial preference of at least one process:

$$VS \;\hat{=}\; (i.decide.(v) \;\Rightarrow\; \exists\, j \cdot j.write.(v,1) \text{ before } i.decide.(v)).$$

From $PS1$, we know that at the earliest a process can decide after rouud 2, and the decision value is always the value last written.

$$i.decide.(v) \;\Rightarrow \exists\, r' > 1 \cdot i.write(v,r') \text{ before } i.decide.(v).$$

Lemma 9.2.1 states that a process can only write a value iu a round r' above round $r > 0$ if at least one process preferred that value in r.

Lemma 9.2.1 $\forall\, i,j \cdot PS_i \wedge RS_j$
$\Rightarrow\; (\exists\, r' > r > 0 \cdot i.write.(v,r') \Rightarrow \exists\, j \cdot j.write.(v,r) \text{ before } i.write.(v,r')).$

□

Proof Suppose that $i.P$ is the first processor to write p in a round r' which is above r:

$$\exists\, i, r' > r \cdot i.write.(v,r') \tag{9.1}$$
$$\wedge\, \forall j \cdot \neg(r'' > r \wedge j.write.(v,r'') \text{ before } i.write.(v,r')). \tag{9.2}$$

Line 9.1 and $PS2_i$ together imply that

$$\exists\, \boldsymbol{v},\boldsymbol{r} \cdot i.write.(v,r') \text{ after } i.Read(\boldsymbol{v},\boldsymbol{r}) \wedge r_i \geq r \wedge \neg leaders\_agree(1-v,\boldsymbol{v},\boldsymbol{r}).$$

If the leaders do not agree on $1-v$ at least one leader prefers p. Also since $i.P$ has already reached round r the leaders must have at least round number r, giving

$$\exists j, r'' \geq r > 0 \cdot i.j.read(v,r'') \text{ before } i.write.(v,r').$$

By RS_j the fact that $i.P$ has read (v,r'') from the j^{th} register and $r'' \geq r > 0$ means that process $j.P$ wrote these values beforehand:

$$\exists j, r'' \geq r > 0 \cdot j.write.(v,r'') \text{ before } i.write.(v,r').$$

Since $i.P$ is the first process to write v in a round strictly above r (line 9.2) it follows that $r'' = r$, i.e.

$$\exists j \cdot j.write.(v,r) \text{ before } i.write.(v,r').$$

□

Lemma 9.2.1 implies that if a process writes v in round r' then in every round below r' from round 1 upwards at least one process must also have written v. Thus

$$\forall i, j \cdot PS_i \wedge RS_j \Rightarrow VS.$$

Hence the protocol is valid:

$$CP \textbf{ sat } VS.$$

To prove that the protocol is consistent we use two corollaries of lemma 9.2.1. Firstly, the contrapositive of lemma 9.2.1 implies that if all processes that complete round r prefer the same value then all processes that complete a higher round also prefer that value. (This equivalent to saying that if no processor prefers v in round r then no processor prefers v in a round above r, which is the form we use in the corollary).

Corollary 9.2.2 $\forall i, j \cdot PS_i \wedge RS_j$
$\Rightarrow (\neg \exists i \cdot i.write.(v, r) \Rightarrow \neg \exists i, r' > r \cdot i.write.(v, r'))$. □

Secondly, lemma 9.2.1 implies that the first processor to write v in round r does so before any processor can write v in a higher round.

Corollary 9.2.3 $\forall i, j \cdot PS_i \wedge RS_j$
$\Rightarrow$ $(i.write.(v, r) \wedge \neg \exists k \cdot k.write.(v, r)$ **before** $i.write.(v, r)$
$\Rightarrow \forall r' > r \cdot i.write.(v, r)$ **before** $j.write.(v, r'))$. □

The protocol is consistent if CP **sat** CS where

$$CS \triangleq \neg(i.decide.(v) \wedge j.decide.(1-v)).$$

In the next lemma we show that if a process decides v in round $r+1$ then all processes which complete round r prefer v (even if they reach r only after the first process decided).

Lemma 9.2.4 $\forall i, j \cdot PS_i \wedge RS_j$
$\Rightarrow$ $(i.decide.(v)$ **after** $i.Read(v, r) \wedge r_i = r \wedge \neg \exists v, r' > r \cdot i.write.(v, r')$
$\Rightarrow \neg \exists j \cdot j.write.(1 - v, r))$.
□

Proof The hypothesis and $PS1$, together imply that

$$i.Read(v, r) \wedge r_i = r \wedge v_i = v \wedge i.can\_decide(v, r). \tag{9.3}$$

If the conclusion is false there must be a processor $j.P$ which was the first to prefer $1-v$ in round r.

$$\exists\, j \cdot j.write.(1-v,r) \wedge \neg\exists\, k \cdot k.write.(1-v,r) \text{ before } j.write.(1-v,r). \quad (9.4)$$

This processor could have preferred either value in round $r-1$. Suppose it preferred $1-v$ in round $r-1$. Then line 9.3 implies

$$i.write(v,r) \text{ before } i.j.read(v_j,r_j) \text{ before } j.write(1-v,r-1).$$

because reading $(1-v,r-1)$ from the j^{th} register would have prevented $i.P$ from deciding. Also, by $PS2_j$ and RS_j,

$$\exists\, \boldsymbol{v}', \boldsymbol{r}' \cdot j.write(1-v,r-1) \text{ before } j.Read(\boldsymbol{v}',\boldsymbol{r}') \text{ before } j.write(1-v,r)$$
$$\wedge\ v_i' = v \wedge r_i' = r \wedge v_j' = 1-v \wedge r_j' = r-1$$

i.e. for $j.P$ to proceed to round r it must read the registers after writing $1-v$ in round $r-1$. Therefore it must see that $i.P$ prefers v in round r. Since by assumption $j.P$ is the first process to prefer $1-v$ in round r (or higher, by corollary 9.2.3), it must see that the leaders prefer v. But then it cannot write $1-v$ in round r, contradicting the assumption (9.4). Suppose therefore that $j.P$ preferred v in round $r-1$.

$$\exists\, \boldsymbol{v}', \boldsymbol{r}' \cdot j.write(v,r-1) \text{ before } j.Read(\boldsymbol{v}',\boldsymbol{r}') \text{ before } j.write(1-v,r)$$
$$\wedge\ v_i' = v \wedge r_i' = r \wedge v_j' = v \wedge r_j' = r-1.$$

By $PS2_j$ it could switch to preferring $1-v$ in round r only if it saw the leaders prefer $1-v$. Then the leaders could at most have roundnumber $r-1$ since by assumption $j.P$ is the first process to prefer $1-v$ in round r (or higher, by corollary 9.2.3). But $j.P$ itself is already at round $r-1$ and prefers v. Thus it cannot see the leaders prefer $1-v$, again contradicting the assumption (9.4).

It follows that it is impossible for $j.P$ to prefer $1-v$ in round r. □

Lemma 9.2.4 and corollary 9.2.2 imply that if a processor decides v in round $r+1$, say, then all processors prefer v in round r and all higher rounds. By $PS1_i$, a processor can only decide the value it last wrote. Cousistency follows.

Liveness

It remains to prove that every processor which is given the opportunity to take infinitely many steps will eventually decide. We first show that if the processors agree for the first time in round r then they decide at most two rounds later.

Lemma 9.2.5 $\forall\, i,j \cdot PS_i \wedge RS_j$
$$\Rightarrow\ \exists\, i \cdot i.write.(v,r) \wedge \neg\exists\, j \cdot j.write.(1-v,r)$$
$$\Rightarrow \neg\exists\, i \cdot (\exists\, q \cdot i.write.(q,r+2) \vee i.decide.(1-v)).$$ □

Proof Suppose that the conclusion is false, i.e. that the process does get to write a value in round $r+2$ or decides on $1-v$. By corollary 9.2.2 the assumption $\neg \exists j \cdot j.write.(1-v, r)$ implies that no process can write $1-v$ in or above round r. Therefore the only possible decision is v. Thus we are left with the possibility that a processor reaches round $r+2$ without deciding. Let $i.write.(v, r+2)$ be the first write in round $r+2$. We know from $PS3_i$ that $i.P$ cannot have decided before this round. So we have

$$\exists i \cdot i.write.(v, r+2) \wedge$$
$$\neg \exists k \cdot k.write.(v, r+2) \text{ before } i.write.(v, r+2) \wedge$$
$$\neg \exists q \cdot i.decide.(q) \text{ before } i.write.(v, r+2).$$

Together with PS, this implies that

$$\exists i, v, r \cdot i.write.(v, r+2) \text{ after } i.Read(v, r) \wedge$$
$$\neg i.can\_decide(v, r) \wedge \forall k \cdot r_k \leq r+1 \wedge r_i = r+1.$$

Expanding the last line gives

$$\exists j \cdot (r_j > r_i \wedge (r_j > r_i + 1 \vee v_j \neq v)) \wedge \forall k \cdot r_k \leq r+1 \wedge r_i = r+1$$

So we are left with $\exists j \cdot r \leq r_j \leq r+1 \wedge v_j \neq v$. But this contradicts the fact that at and above round r all processes prefer v. Hence no process can write a value in round $r+2$. □

Thus to prove that the protocol terminates we only need to show that it will eventually get to a round where all processes agree. This depends to some extent on how the processors are interleaved. An interleaving which can take account of the state of the processors can force the protocol to continue forever, as in figure 9.2.

This strategy works only if the interleaving can take account of the processor's decision whether or not to advance its roundnumber upon observing disagreement. This is not the case if the interleaving of the processors is independent of the choice of event by the processors. More formally we assume that the protocol runs in an environment D such that if a, b are two events in the alphabet B_i of the i^{th} processor then for all $0 \leq j < N$

$$[\![CP \parallel D]\!](\{u \mid u_{n+1} \in B_j\} | \{u \mid u_n = a\})$$
$$= [\![CP \parallel D]\!](\{u \mid u_{n+1} \in B_j\} | \{u \mid u_n = b\})$$

Ideally we would like to prove that $[\![CP \parallel D]\!]\ TS = 1$ for any D which satisfies the independence assumption. However, as in the proof of the original algorithm, we shall be limited to a worst case argument. If we assume that the scheduling is independent of the state of the processors the best strategy of the scheduler to

$\langle \ \ldots$	Suppose that $j.P$ and $k.P$ are the only leaders in round
$k.write.(1, r-1)$,	$r-1$ and that they disagree. Suppose also that $k.P$
$k.j.read.(0, r-2)$,	reads $j.R$ before $j.P$ writes to it in round $r-1$. Then
$k.k.read.(1, r-1)$,	$k.P$ believes itself to be the only leader. Its next step will be a write with the same preference and an increased roundnumber.
$j.write.(0, r-1)$,	Now $j.P$ is allowed to write its $r-1$-round preference
$j.k.read.(1, r-1)$,	and read the registers. $j.P$ observes disagreement and
$j.j.read.(0, r-1)$,	may decide to stay in round $r-1$, but if it is forced to
$j.write.(0, r-1)$,	go on reading it will eventually advance to round r.
$\vdots$	
$j.write.(0, r)$,	
$j.k.read.(1, r-1)$,	Now it is allowed one more reading which will lead it to
$j.j.read.(0, r)$,	believe that it is the only leader in round r. Letting $k.P$
$k.write.(1, r)$	do its write for round r will restart the procedure.
$\ldots \quad \rangle$	

Figure 9.2: A non-terminating interleaving

delay termination is to choose all processes equally often. Otherwise a subset of the processors is going to get ahead of the others and become leaders. The fewer leaders there are the more likely they are to agree. We therefore suppose that the processors run in lockstep.

$$\begin{aligned} D &= D_0 \\ D_i &= a : B_i \to D_{i \oplus 1} \end{aligned}$$

where $\oplus$ denotes addition modulo N. It can be shown (using recursion induction) that the parallel system $CP \parallel D$ contains no external choice. It can therefore be analysed as a probability measure on the space of infinite traces.

With lockstep interleaving, every processor takes one reading of the registers and does one write in every $N(N+1)$ steps of the protocol. Let $N(N+1)$ steps be a cycle. Let $A(n)$ denote the set of traces such that in the n^{th} cycle the leaders have reached agreement.

$$\begin{aligned} A(n) = \{u \mid \exists\, r, p, i \cdot i.write.(v, r)(u \restriction (nN(N+1))) \\ \wedge \neg \exists\, j \cdot j.write.(v, r)(u \restriction (nN(N+1)))\}. \end{aligned}$$

The probability that the processes will eventually reach agreement can be calculated as

$$[\![CP \| D]\!] TS$$
$$= \sum_{n=1}^{\infty} \left([\![CP \| D]\!] \, (A(n) \mid A(n-1)^c) \prod_{i=1}^{n-1} [\![CP \| D]\!] \, (A(i)^c \mid A(i-1)^c) \right).$$

If there is no agreement in the $(n-1)^{\text{th}}$ cycle then there must be at least two leaders with differing preferences. Let l be the number of leaders and let k be the number of leaders whose preference is 1. The probability of agreement in the n^{th} cycle given disagreement in the $(n-1)^{\text{th}}$ cycle can be calculated as the probability that at least one of the leaders preferring 1 gets ahead and none of the others, which is $(1-p^k)p^{l-k}$, or vice versa. Now

$$\begin{aligned}(1-p^k)p^{l-k} + p^k(1-p^{l-k}) &= p^k - 2p^l + p^{l-k} \\ &\geq 2p^{l/2} - 2p^l \\ &\geq 2p^{N/2} - 2p^N.\end{aligned}$$

The last step holds only if $l \geq 2\log_p(1/2)$ but if $p \geq 1/2$ this is always true. This gives us a lower bound for $[\![CP \| D]\!](A(n) \mid A(n-1)^c)$. It follows that the real value of $[\![CP \| D]\!](A(n) \mid A(n-1)^c)$, δ say, is a strictly positive quantity. Hence

$$\begin{aligned}[\![CP \| D]\!] TS &= \sum_n (1-\delta)^n \delta \\ &= 1.\end{aligned}$$

We can now choose p so as to maximise the lower bound for δ:

$$p = 2^{-2/N}$$

giving $\delta \geq 1/2$. Then the expected number of cycles to reach agreement, $1/\delta$, is less than 2 and the expected number of steps to reach agreement is $\mathcal{O}(n^2)$. This concludes our analysis.

9.3 Discussion

We have used the case study of a randomised consensus protocol to demonstrate the applicability of the process algebra and proof rules which we developed in the earlier chapters. This has been successful because the process algebra has proved expressive enough to capture the algorithm which implements the protocol, and the proof rules have been sufficient to enable us to give a formal proof of correctness for the safety properties of the protocol. For the liveness properties we have had to take recourse to a slightly informal worst-case argument. It may be possible

to give a proof of termination which would be valid for any scheduler rather than just the lockstep scheduler, but it would be much more complicated. However, the worst-case argument has been sufficient to show that our formalism is capable of addressing all the issues involved in reasoning about randomised algorithms.

The semantics of the original algorithm were given in terms of I/O automata [LM87]. The proof of correctness was informal, but it contained essentially the same arguments which we used, too. The differences are due mainly to the fact that our algorithm is simpler. It is also faster than the original algorithm, which has a worst-case running time of $\mathcal{O}(2^n)$ steps, and also better than the original algorithm combined with the weak shared coin protocol described in [AH90], which has an expected running time of $\mathcal{O}(n^4)$ steps. One has to bear in mind that this speed-up and simplification is achieved at the cost of guaranteeing termination only for a scheduler which cannot take advantage of the state of the processors. We feel that the simplification and increased efficiency justify this reasonable assumption.

Chapter 10

Discussion

10.1 Conclusions

In this thesis we have presented a mathematical formalism for the specification and proof of correctness of probabilistic communicating processes. We have defined a process algebra which is based on *CSP*, the main difference being that probabilistic choice is substituted for non-deterministic choice. We have given a semantics in terms of probability measures on the space of infinite traces for a model which contains probabilistic choice and all other *CSP*-operators except external choice and alphabetised parallel composition. We have shown that this semantics preserves all the algebraic laws which hold in other models of *CSP*.

To define the semantics of recursion we have used two metrics on the space of probability measures on infinite traces. Convergence with respect to the first metric is equivalent to weak convergence of probability measures, and we have used it to show that under certain conditions, recursive definitions containing unguarded variables as branches of a probabilistic choice are well-defined. Convergence with respect to the second metric implies weak convergence. The second metric is an ultra-metric like the ones used in other models of *CSP* and has allowed us to show that any guarded recursion involving parallel or sequential composition is well-defined.

We have given examples to show how this model enables us to reason about the properties of probabilistic processes, especially liveness properties such as the asymptotic frequency of events.

To be able to reason effectively about concurrency in general, we have defined a semantics for a second model in terms of conditional probability measures. This model contains operators for external choice and alphabetised parallel composition, but not for sequential composition, hiding or interleaving. Like the first model it preserves all the laws for the operators it contains. Again, we have defined recursive

processes by taking recourse to two metrics.

We have given proof rules to relate the process algebra to more abstract specifications defined in terms of predicates upon infinite traces. We have proved each rule to be sound. This has enabled us to reason about safety properties. To reason about liveness properties it is necessary to make some assumptions about the environment of a system, namely that the environment does not block the system and that it resolves all the external choices on which the system depends. We have shown that given such an environment, the resulting system has a well-defined semantics both in the first and in the second model, which means the techniques we used to analyse liveness properties in the first model are also applicable to systems specified in the second model.

We have demonstrated the usefulness of our approach by giving formal treatments of a self-stabilising tokenring and of a randomised consensus protocol.

10.2 Related Work

There exists several formalisms for the specification of probabilistic processes, reflecting the variety of formal methods in general. Broadly speaking, all probabilistic languages define the semantics of choice and parallel composition in terms of sums and products of probabilities respectively. Differences arise in the treatment of external choice and unsynchronised parallel composition, as well as in the methods of defining fixed points and equivalences between processes.

Glabbeek et. al. [GSST90] present three semantic models for PCCS, a probabilistic dialect of Milner's SCCS [Mi89]. The semantics of these models are based on probabilistic labeled transition systems, which are essentially state transition systems with probabilities attached to each branch. Differences between the models arise from the treatment of choice: in the 'reactive' model the probabilities of all transitions with the same action sum to 1, whereas in the 'generative' model the probabilities for all transitions sum to 1. The former can be understood as a mixture of internal and external choice, in the sense that the choice of action is made externally but the choice of transition with a given action is made internally. Parallel composition is defined as lockstep interleaving: a transition in a parallel system is labelled by a pair of actions (with the product of their individual probabilities). They can happen in either order, but both must happen before the next transition. This seems a very restrictive view of parallel composition. Equivalence between processes is established by probabilistic bisimulation (due to Larsen and Skou [LS89]), which is an analog of strong bisimulation. This leads to very fine distinctions between processes; for instance it rules out the law of distributivity of probabilistic choice over prefixing. We think that these distinctions are unnecessarily strong.

Jou and Smolka [JS90] investigate weaker concepts of process equivalence for the generative model. Nearest to PCSP is their concept of trace equivalence, which means that for two processes P and Q each transition path has the same probability, whether it starts at P or at Q. However, this kind of equivalence is not a congruence, i.e. P and Q are not necessarily interchangeable in any expression. The paper also presents a sound and complete axiomatisation of finite serial processes in the generative model with respect to probabilistic bisimulation. The only laws that hold for probabilistic choice are symmetry, associativity and idempotence. As in PCSP, every guarded recursive call has a well-defined fixed point, bound variables can be substituted for, and the unfolding of recursive calls preserves equivalence. There are no laws for parallel composition.

Jones and Plotkin [JP89] aim to provide a general framework for the semantics of probabilistic programming languages, which they base on evaluations. These are functions similar to probability measures but defined only on open sets rather than the general Borel sets. Unlike probability measures, evaluations can easily be partially ordered and can therefore be used to construct a probabilistic powerdomain, $\mathcal{E}(P)$. The authors show that the structure associated with $\mathcal{E}(P)$ is a monad and that recursive domain equations involving $\mathcal{E}(P)$ can be solved in a categorical setting. They then present the semantics of a probabilistic programming language consisting of atomic commands, sequential composition, if-statements, while-loops, probabilistic choice and parallel composition. The latter is parametrised on a probabilistic scheduler which decides, given a state, which process runs next. There is no construct for input or external choice. Thus the expressiveness of their language is about the same as that in the simple model which we presented in chapter 3.

Rao[Rao90] presents a probabilistic extension to UNITY[CM88]. He introduces probabilistic assignment, which probabilistically chooses one of a list of a finite number of possible expressions to assign to a variable. The probability with which an expression is chosen is arbitrary and cannot be made explicit. The only probabilistic property important for Rao is that in an infinite trace of executions of a probabilistic assignment each expression will be chosen infinitely often. He defines the weakest precondition of the probabilistic assignment as the one which holds of every branch. This enables him to extend the usual UNITY proof rules for safety properties to probabilistic programs. He then defines the weakest *probabilistic* precondition as one which must hold of at least one branch and uses it to develop a set of proof rules for liveness properties, which hold with probability 1. His approach is closest to own in that he also uses infinite traces and constructs separate proof rules for safety and liveness properties. However, we think that to throw away any possibility of reasoning about specific probabilities is unnecessarily restrictive. For instance, it means that probabilistic UNITY cannot be used to prove that the probability of return to the origin in a random walk is 1, because this is true if the probability with which a step is made in either direction is $1/2$, but not otherwise.

Also, even where it can be shown that a property is achieved witb probability 1, it may be desirable to calculate something like the expected number of executions until it is achieved, which is impossible without using explicit probabilities. Apart from that, differences between Rao's approach and ours reflect the general differences between UNITY and *CSP*. For instance, the consensus protocol could be specified in terms of probabilistic assignments rather that in terms of communications over channels, though of course control over the probabilities would be lost. Also, UNITY does not allow compositional proofs for concurrent systems in the way that *CSP* does.

10.3 Future Work

There are some questions yet to be investigated regarding the models presented in this thesis, notably whether the laws and proof rules are complete, and the precise relation of these models to other models of *CSP*.

It would be nice to have a semantics for a probabilistic model which contained the full range of *CSP* operators, including probabilistic choice, external choice, general parallel composition, sequential composition and hiding. For the reasons given in chapter 3 it is not possible to express external choice and general parallel composition in terms of probability measures, and for the reasons given in chapter 7 it is not possihle to give a semantics to sequential composition and hiding in terms of conditional prohability measure. One would therefore have to look at entirely different semantics to the ones considered here.

There is also a nced for an entirely different prohabilistic model, in which the probability concerns not the choice of action, but the time at which it happens. Such a model would for instance address the probabilistic aspects of the Ethernet protocol which are lcft out of the formal specification presented by Davies[Dav91]. Work in this direction has already begun [SNH92].

Bibliography

[AH90] J. Aspnes and M. Herlihy, *Fast Randomized Consensus Using Shared Memory*, J. Algorithms, 11 (1990), 441-461.

[AS85] B. Alpern and F.B. Schneider, *Defining Liveness*, Inf. Proc. Letters 21 (1985), 181-185.

[Bi79] P. Billingsley, *Probability and Measure*, Wiley, 1979.

[CM88] K.M. Chandy and J. Misra, *Parallel Program Design: A Foundation*, Addison-Wesley, 1988.

[Ch90] I. Christoff, *Testing Equivalences and Fully Abstract Models for Probabilistic Processes*, Concur 90, Theories of Concurrency, Unification and Extension, Springer Verlag LNCS 458 (1990).

[Dav91] J. Davies, *Specification and Proof in Real-time Systems*, Oxford University D.Phil thesis 1991.

[DS91] J. Davies, S. Schneider, *Recursion Induction for Real-time Processes*, submitted for publication.

[Fel57] W. Feller, *An Introduction to Probability Theory and its Applications*, volume I, Wiley, 2 edition, 1957.

[GSST90] R.J. van Glabbeek, S.A. Smolka, B. Steffen and C. Tofts, *Reactive, Generative and Stratified Models of Probabilistic Processes*, IEEE Symp. on Logic in Computer Science, Philadelphia, PA., USA, June 1990.

[Herm90] T. Herman, *Probabilistic Self-Stabilization*, Inf. Proc. Letters 35 (1990), 63-67.

[Hoa85] C.A.R. Hoare, *Communicating Sequential Processes*, Prentice-Hall, 1985.

[JP89] C. Jones and G. Plotkin, *A Probabilistic Powerdomain of Evaluations*, Proceedings of 4th Annual Symposium on Logic in Computer Science, 1989.

[JS90] C. Jou and S.A. Smolka, *Equivalences, Congruences, and Complete Axiomatizations for Probabilistic Processes*, Concur 90, Theories of Concurrency, Unification and Extension, Springer Verlag LNCS 458 (1990).

[LS89] K.G. Larsen and A. Skou, *Bisimulation through Probabilistic Testing*, Proceedings of 16th ACM Symp. on Principles of Programming Languages, Austin, TX (1989).

[LM87] N.A. Lynch and M. Merritt, *Introduction to the Theory of Nested Transactions*, Technical Report MIT/LCS/TR-387, MIT Laboratory for Computer Science, April 1986.

[Mi89] R. Milner, *Communication and Concurrency*, Prentice-Hall, 1989.

[Pnu83] A. Pnueli, On the Extremely Fair Treatment of Probabilistic Algorithms, *Proceedings of the 15th Annual Symposium on the Theory of Computing*, (1983), 278-290.

[Pu90] W. Pugh, *Skip Lists: A Probabilistic Alternative to Balanced Trees*, Communications of the ACM. June 1990, Vol. 33, 668-676.

[Rao90] J.R. Rao, *Reasoning about Probabilistic Parallel Programs*, submitted to ACM Trans. on Programming Languages and Systems.

[Re88] G.M. Reed, *A Uniform Mathematical Theory for Real-time Distributed Computing*, Oxford University D.Phil thesis 1988.

[Ros82] A.W. Roscoe, *A Mathematical Theory of Communicating Processes*, Oxford University D.Phil thesis 1982.

[Ros88] A.W. Roscoe, *Unbounded Nondeterminism in CSP*, Technical Monograph PRG-67, 27-80, July 1988.

[ReR88] G.M. Reed and A.W.Roscoe, *A Timed Model for Communicating Sequential Processes*, Proceedings of ICALP'86, Springer LNCS 226 (1986), 314-323; Theoretical Computer Science 58 (1988), 249-261.

[Sh84] A.N. Shiryayev, *Probability*, Springer Verlag, 1984.

[SS90] S.A. Smolka and B. Steffen, *Priority as Extremal Probability*, Concur 90, Theories of Concurrency, Unification and Extension, Springer Verlag LNCS 458 (1990).

[SNH92] E.V. Sørenseu, J. Nordahl and N.H. Hansen, *From CSP Models to Markov Models*, To appear in IEEE transactions on Software Engineering.

[Su75] W.A. Sutherland, *Introduction to Metric and Topological Spaces*, Oxford University Press, 1975.