Basic Research in Computer Science

A Linear Metalanguage for Concurrency

Glynn Winskel

BRICS Report Series **RS-98-31**

ISSN 0909-0878 **November 1998**

See back inner page for a list of recent BRICS Report Series publications. Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark
Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/98/31/`

A Linear Metalanguage for Concurrency*

Glynn Winskel

BRICS**, University of Aarhus, Denmark

Abstract. A metalanguage for concurrent process languages is introduced. Within it a range of process languages can be defined, including higher-order process languages where processes are passed and received as arguments. (The process language has, however, to be linear, in the sense that a process received as an argument can be run at most once, and not include name generation as in the Pi-Calculus.) The metalanguage is provided with two interpretations both of which can be understood as categorical models of a variant of linear logic. One interpretation is in a simple category of nondeterministic domains; here a process will denote its set of traces. The other interpretation, obtained by direct analogy with the nondeterministic domains, is in a category of presheaf categories; the nondeterministic branching behaviour of a process is captured in its denotation as a presheaf. Every presheaf category possesses a notion of (open-map) bisimulation, preserved by terms of the metalanguage. The conclusion summarises open problems and lines of future work.

1 Introduction

Over the last few years, Gian Luca Cattani and I have worked on presheaf models for interacting processes, culminating in Cattani's forthcoming PhD thesis [3]. The work started from the general definition of bisimulation via open maps in [13] which suggested examining a broad class of models for concurrency—presheaf categories. Later we realised that presheaf models can themselves be usefully assembled together in a category in which the maps are colimit-preserving functors. There are two main benefits: one is a general result stating that colimit-preserving functors between presheaf categories preserve open maps and bisimulation [6]; the other that the category of the presheaf models is a form of domain theory for concurrency, with a compositional account of bisimulation, though at the cost that domains are categories rather than special partial orders [20, 4].

* Invited paper AMAST '98, Brazil

** **B**asic **R**esearch **i**n **CS**, Centre of the Danish National Research Foundation.

We originally concentrated on the category of presheaf categories with colimit-preserving functors (or equivalently, the bicategory of profunctors). We've come to realise that by shifting category, to presheaf categories with *connected*-colimit preserving functors, a lot of our work can be done more systematically. (A connected colimit is a colimit of a nonempty connected diagram.) In particular the new category supports a metalanguage in which many of our constructions can be defined once and for all. This is not the only way the metalanguage saves work. Its terms will automatically preserve connected colimits. The metalanguage supports recursive definitions because ω-colimits are examples of connected colimits. Connected-colimit preserving functors preserve open-map bisimulation. Consequently terms of the metalanguage preserve open-map bisimulation; if two terms which are open-map bisimilar are substituted for the same variable in a term of the metalanguage then the resulting terms will be open-map bisimilar.

The metalanguage can be interpreted in a wide range of categories. To spare some of the overhead of working with presheaf categories the metalanguage will first be interpreted in a simple category of nondeterministic domains. Equality of terms in this model will coincide with trace equivalence. However the nondeterministic domains are mathematically close to presheaf categories. With a switch of viewpoint, essentially the same constructions lead to an interpretation of the metalanguage in presheaf categories with connected-colimit preserving functors, for which open-map bisimulation is an appropriate equivalence.

2 Presheaf models sketched

Let $\mathbb{P}$ be a small category. The category of *presheaves over* $\mathbb{P}$, written $\widehat{\mathbb{P}}$, is the category $[\mathbb{P}^{op}, \mathbf{Set}]$ with objects the functors from $\mathbb{P}^{op}$ (the opposite category) to **Set** (the category of sets and functions) and maps the natural transformations between them.

In our applications, the category $\mathbb{P}$ is thought of as consisting of abstract paths (or computation-path shapes) where a map $e : p \to p'$ expresses how the path p is extended to the path p'. In this paper the categories over which we take presheaves will be (the category presentation of) partial orders; the way a path p extends to a path

p' will be unique and a map from a path p to a path p' simply a witness to $p \leq p'$ in the partial order.

A presheaf $X : \mathbb{P}^{op} \to \mathbf{Set}$ specifies for a typical path p the set $X(p)$ of computation paths of shape p. The presheaf X acts on $e : p \to p'$ in $\mathbb{P}$ to give a function $X(e)$ saying how p'-paths in X restrict to p-paths in X. In this way a presheaf can model the nondeterministic branching of a process.

Bisimulation on presheaves is derived from notion of open map between presheaves [12, 13]. Open maps are a generalisation of functional bisimulations, or zig-zag morphisms, known from transition systems [13]. Presheaves in $\widehat{\mathbb{P}}$ are *bisimilar* iff there is a span of surjective (i.e., epi) open maps between them.

Because the category of presheaves $\widehat{\mathbb{P}}$ is characterised abstractly as the free colimit completion of $\mathbb{P}$ we expect that colimit-preserving functors between presheaf categories to be useful. They are, but not all operations associated with process languages preserve arbitrary colimits. Prefixing operations only preserve connected colimits while parallel compositions usually only preserve connected colimits in each argument separately. However, the preservation of connected colimits is all we need of a functor between presheaf categories for it to preserve bisimulation.

Proposition 1. *[7] Let $G : \widehat{\mathbb{P}} \to \widehat{\mathbb{Q}}$ be any connected-colimit preserving functor between presheaf categories. Then G preserves surjective open maps and open-map bisimulation.*

Define $\mathcal{C}\mathbf{on}$ to be the category consisting of objects partial orders $\mathbb{P}, \mathbb{Q}, \cdots$, with maps $g : \mathbb{P} \to \mathbb{Q}$ the connected-colimit preserving functors $g : \widehat{\mathbb{P}} \to \widehat{\mathbb{Q}}$ between the associated presheaf categories, and composition the usual composition of functors. Define $\mathcal{C}\mathbf{ol}$ to be the subcategory of colimit-preserving functors.

3 Categories of nondeterministic domains

We obtain nondeterministic domains by imitating the definitions on presheaves but replacing **Set** by the much simpler partial order category **2** with two elements $\mathbf{0}, \mathbf{1}$ ordered by $\mathbf{0} \leq \mathbf{1}$.

Instead of presheaves $\widehat{\mathbb{P}} = [\mathbb{P}^{op}, \mathbf{Set}]$ we now obtain $\widehat{\mathbb{P}} = [\mathbb{P}^{op}, \mathbf{2}]$, functors, and so monotonic functions from $\mathbb{P}^{op}$ to **2**. It's not hard

to see that an object x of $\widehat{\mathbb{P}}$ corresponds to a downwards-closed set given by $\{p \in \mathbb{P} \mid x(p) = \mathbf{1}\}$, and that a natural transformation from x to y in $\widehat{\mathbb{P}}$ corresponds to the inclusion of $\{p \in \mathbb{P} \mid x(p) = \mathbf{1}\}$ in $\{p \in \mathbb{P} \mid y(p) = \mathbf{1}\}$. So we can identify $\widehat{\mathbb{P}}$ with the partial order of downwards-closed subsets of $\mathbb{P}$, ordered by inclusion. Thought of in this way it is sensible to think of $\widehat{\mathbb{P}}$ as a nondeterministic domain in the sense of [10, 9]; the order $\widehat{\mathbb{P}}$ has joins got simply via unions so it is certainly a cpo, with least element $\emptyset$, and we can think of the union operation as being a form of nondeterministic sum. It's worth remarking that the domains obtained in this way are precisely the infinitely-distributive algebraic lattices (see e.g. [18, 19]) and that these are just the same as the prime algebraic lattices of [17], and free join completions of partial orders.

There are several choices about what to take as maps between nondeterministic domains. If we eschew "fairness", the most generous we seem to have call for is that of all Scott-continuous functions between the domains. We are interested in maps which are just broad enough to include those operations we associate with interacting processes, operations such prefixing of actions, nondeterministic sum and parallel composition, so we look for a narrower class of maps than continuous functions.

3.1 The category $\mathcal{D}\mathbf{om_s}$

On mathematical grounds it is natural to consider taking maps between nondeterministic domains which preserve their join structure, to choose functions f from $\widehat{\mathbb{P}}$ to $\widehat{\mathbb{Q}}$ which preserve all joins, i.e. so $f(\bigcup X) = \bigcup_{x \in X} f(x)$. Such functions (known often as *additive* functions) compose as usual, have identities and give rise to a category rich in structure. Call this category $\mathcal{D}\mathbf{om_s}$ and write $f : \mathbb{P} \to_s \mathbb{Q}$ for a map in $\mathcal{D}\mathbf{om_s}$, standing for an additive function from $\widehat{\mathbb{P}}$ to $\widehat{\mathbb{Q}}$. Notice that such maps can be presented in several different ways. Because such maps preserve joins they are determined by their results on just the "complete primes", elements $p{\downarrow} \in \widehat{\mathbb{P}}$, for $p \in \mathbb{P}$, such that

$$p{\downarrow}(p') = \mathbf{1} \text{ if } p' \leq p, \text{ and } \mathbf{0} \text{ otherwise.}$$

Let $f : \mathbb{P} \to_s \mathbb{Q}$, so $f : \widehat{\mathbb{P}} \to \widehat{\mathbb{Q}}$, and write $f^o : \mathbb{P} \to \widehat{\mathbb{Q}}$ for its restriction such that $f^o(p) = f(p{\downarrow})$, for $p \in \mathbb{P}$. As every element x

of $\widehat{\mathbb{P}}$ is the join $\bigcup_{p\in x} p{\downarrow}$ we see that

$$f(x) = \bigcup_{p\in x} f^o(p) \ .$$

In this way maps $f : \mathbb{P} \to_s \mathbb{Q}$ correspond to monotonic functions $f^o : \mathbb{P} \to \widehat{\mathbb{Q}}$. But monotonic functions $g : \mathbb{P} \to \widehat{\mathbb{Q}}$ are just the same as monotonic functions $g : \mathbb{P} \to [\mathbb{Q}^{op}, \mathbf{2}]$ and, uncurrying, these correspond to monotonic functions $h : \mathbb{P} \times \mathbb{Q}^{op} \to \mathbf{2}$ and so to elements of $\widehat{\mathbb{P}^{op} \times \mathbb{Q}} = [(\mathbb{P}^{op} \times \mathbb{Q})^{op}, \mathbf{2}]$. This suggests that $\mathbb{P}^{op} \times \mathbb{Q}$ is a function space, as indeed is so.

The category $\mathcal{D}\mathbf{om_s}$ is monoidal-closed and in fact carries enough structure to be a categorical model of classical linear logic, the involution of linear logic, $\mathbb{P}^{\perp}$, being given as $\mathbb{P}^{op}$. The tensor of $\mathbb{P}$ and $\mathbb{Q}$ is given by the product of partial orders $\mathbb{P} \times \mathbb{Q}$ and the function space from $\mathbb{P}$ to $\mathbb{Q}$ by $\mathbb{P}^{op} \times \mathbb{Q}$. Its products and coproducts are both given by disjoint unions on objects; for example the usual product of domains $\widehat{\mathbb{P}} \times \widehat{\mathbb{Q}}$ is easily seen to be isomorphic to $\widehat{\mathbb{P} + \mathbb{Q}}$, the nondeterministic domain of the disjoint union $\mathbb{P} + \mathbb{Q}$.

3.2 Lifting

One important construction on domains, that of lifting, is missing. Lifting a domain places a new bottom element below a domain. We can achieve this by adjoining a new element $\perp$ below a copy of $\mathbb{P}$ to obtain $\mathbb{P}_{\perp}$; a way to realise this is by taking $\perp$ to be the empty set $\emptyset$ and the copy of $\mathbb{P}$ to be $\{p{\downarrow} \mid p \in \mathbb{P}\}$ so that the order of $\mathbb{P}_{\perp}$ is given simply by restricting the order of $\widehat{\mathbb{P}}$. Operations on processes, notably prefixing and parallel composition, make essential use of an operation associated with lifting. The operation is the function

$$\lfloor - \rfloor : \widehat{\mathbb{P}} \to \widehat{\mathbb{P}_{\perp}}$$

such that $\lfloor x \rfloor(\perp) = \mathbf{1}$ and $\lfloor x \rfloor(p{\downarrow}) = x(p)$ for $x \in \widehat{\mathbb{P}}$. But the function $\lfloor - \rfloor$ is not a map from $\mathbb{P}$ to $\mathbb{P}_{\perp}$ in $\mathcal{D}\mathbf{om_s}$ as it does not preserve all joins; the problem occurs with the join of the empty set $\bigcup \emptyset$, the least element of $\widehat{\mathbb{P}}$, which is not sent to the least element of $\widehat{\mathbb{P}_{\perp}}$.

3.3 The category $\mathcal{D}\mathbf{om}$

To accommodate the function $\lfloor - \rfloor$ we are forced to move to a slightly broader category, though fortunately one that inherits a good many properties from $\mathcal{D}\mathbf{om_s}$. The category $\mathcal{D}\mathbf{om}$ has the same objects, partial orders, but its morphisms from $\mathbb{P}$ to $\mathbb{Q}$, written $f : \mathbb{P} \to \mathbb{Q}$, are functions $f : \widehat{\mathbb{P}} \to \widehat{\mathbb{Q}}$ which need only preserve *nonempty* joins, or more accurately, joins of non-empty sets.

Maps $\mathbb{P} \to \mathbb{Q}$ in $\mathcal{D}\mathbf{om}$ are determined by their action on $\emptyset$ and $p{\downarrow}$, for $p \in \mathbb{P}$. This is because any $x \in \widehat{\mathbb{P}}$ is trivially the nonempty join with the least element $\emptyset \cup \bigcup_{p \in x} p{\downarrow}$. Given the way to represent $\mathbb{P}_\perp$ as consisting precisely of the elements $\emptyset$ and $p{\downarrow}$, for $p \in \mathbb{P}$, there is an embedding $j : \mathbb{P}_\perp \to \widehat{\mathbb{P}}$. So any map $f : \mathbb{P} \to \mathbb{Q}$ is determined by its restriction $f \circ j : \mathbb{P}_\perp \to \widehat{\mathbb{Q}}$. The restriction $f \circ j$ is clearly monotonic. Moreover any monotonic function $g : \mathbb{P}_\perp \to \widehat{\mathbb{Q}}$ has an extension[1] $g^\dagger : \mathbb{P} \to \mathbb{Q}$ in $\mathcal{D}\mathbf{om}$ given by $g^\dagger(x) = \bigcup_{p \in \lfloor x \rfloor} g(p{\downarrow})$ for $x \in \widehat{\mathbb{P}}$. The two operations $(-) \circ j$ and $(-)^\dagger$ are mutually inverse. Consequently maps $\mathbb{P} \to \mathbb{Q}$ in $\mathcal{D}\mathbf{om}$ correspond bijectively to maps $\mathbb{P}_\perp \to_s \mathbb{Q}$ in $\mathcal{D}\mathbf{om_s}$,[2] and so to elements in $\widehat{(\mathbb{P}_\perp)^{op} \times \mathbb{Q}}$.

3.4 Fixed points

The set of maps in $\mathcal{D}\mathbf{om}$ from a path order $\mathbb{P}$ to one $\mathbb{Q}$ inherits an order from elements of the function space $\widehat{(\mathbb{P}_\perp)^{op} \times \mathbb{Q}}$. Operations of the category $\mathcal{D}\mathbf{om}$ will come to preserve nonempty joins of such maps and, in particular, joins of ω-chains. Hence operations F of $\mathcal{D}\mathbf{om}$ taking maps $\mathbb{P} \to \mathbb{Q}$ to maps $\mathbb{P} \to \mathbb{Q}$ will have least fixed points $\mathit{fix}\ F : \mathbb{P} \to \mathbb{Q}$.

3.5 Intuition

How is one to think of the category $\mathcal{D}\mathbf{om}$? The interpretation we'll give and the way in which we define denotational semantics to process languages will have some novelty, though similar uses of categories of nondeterministic domains have been made (see for instance

[1] In fact, the left Kan extension along j.

[2] The correspondence is natural in $\mathbb{P}$ and $\mathbb{Q}$ making $\mathcal{D}\mathbf{om}$ the coKliesli category associated to the comonad $(-)_\perp$ on $\mathcal{D}\mathbf{om_s}$ and a reflective subcategory of $\mathcal{D}\mathbf{om_s}$.

[10, 9]). An object $\mathbb{P}$ is to be thought of as consisting of finite computation paths (each one a "trace" in the sense of [11]), for example the finite string of actions that a CCS or CSP process might perform. The partial order $p \leq p'$ on $\mathbb{P}$ is thought of as saying that the computation path p can be extended to the computation path p'. With this intuition in mind we shall call the objects of $\mathcal{D}\mathbf{om}$ *path orders*. An element of $\widehat{\mathbb{P}}$ is a trace set as in [11] and stands for the set of computation paths a nondeterministic process can perform.

A map $f : \mathbb{P} \to \mathbb{Q}$ takes a nondeterministic process with computation paths in $\mathbb{P}$ as input and yields a nonderministic process with computation paths in $\mathbb{Q}$ as output. How is one to understand that a map preserves joins of nonempty sets? Because the map need only preserve nonempty joins it is at liberty to ignore the input process in giving nontrivial output. Because the map preserves all nonempty joins the interaction with the input process has to be conducted in a linear way; the input process cannot be copied to explore its different nondeterministic possibilities, so once started it can only follow a single course of computation, during which it may be interacted with intermittently. It's helpful to think of a map in $\mathcal{D}\mathbf{om}$ as a context which surrounds an input process interacting with the input process occasionally and sometimes interacting with its own environment; whichever computation path the output process (the context surrounding the input process) follows it can only involve the input process following a single computation path.

4 Constructions on path orders

4.1 Tensor

The tensor of path orders $\mathbb{P} \otimes \mathbb{Q}$ is given by the set $(\mathbb{P}_\perp \times \mathbb{Q}_\perp) \setminus \{(\perp, \perp)\}$, ordered coordinatewise, in other words, as the product of $\mathbb{P}_\perp$ and $\mathbb{Q}_\perp$ as partial orders but with the bottom element $(\perp, \perp)$ removed.

Let $f : \mathbb{P} \to \mathbb{P}'$ and $g : \mathbb{Q} \to \mathbb{Q}'$. We define $f \otimes g : \mathbb{P} \otimes \mathbb{Q} \to \mathbb{P}' \otimes \mathbb{Q}'$ as the extension (*cf.* Section 3.3) $h^\dagger$ of a monotonic function

$$h : (\mathbb{P} \otimes \mathbb{Q})_\perp \to \widehat{\mathbb{P}' \otimes \mathbb{Q}'} \ .$$

Notice that $(\mathbb{P} \otimes \mathbb{Q})_\perp$ is isomorphic to the product as partial orders of $\mathbb{P}_\perp \times \mathbb{Q}_\perp$ in which the bottom element is then $(\perp, \perp)$. With this realisation of $(\mathbb{P} \otimes \mathbb{Q})_\perp$ we can define $h : \mathbb{P}_\perp \times \mathbb{Q}_\perp \to \widehat{\mathbb{P}' \otimes \mathbb{Q}'}$ by taking

$$(h(p,q))(p',q') = \lfloor f(p) \rfloor (p') \times \lfloor g(q) \rfloor (q')$$

for $p \in \mathbb{P}_\perp$, $q \in \mathbb{Q}_\perp$ and $(p', q') \in \mathbb{P}' \otimes \mathbb{Q}'$—on the right we use the product, or meet, of $\mathbf{2}$, so $\mathbf{0} \times \mathbf{0} = \mathbf{0} \times \mathbf{1} = \mathbf{1} \times \mathbf{0} = \mathbf{0}$ and $\mathbf{1} \times \mathbf{1} = \mathbf{1}$.

The unit for tensor is the empty path order $\mathbb{O}$.

Elements $x \in \widehat{\mathbb{P}}$ correspond to maps $\tilde{x} : \mathbb{O} \to \mathbb{P}$ sending the empty element to x. Given $x \in \widehat{\mathbb{P}}$ and $y \in \widehat{\mathbb{Q}}$ we define $x \otimes y \in \widehat{\mathbb{P} \otimes \mathbb{Q}}$ to be the element pointed to by $\tilde{x} \otimes \tilde{y} : \mathbb{O} \to \mathbb{P} \otimes \mathbb{Q}$.

4.2 Function space

The function space of path orders $\mathbb{P} \multimap \mathbb{Q}$ is given by the product of partial orders $(\mathbb{P}_\perp)^{op} \times \mathbb{Q}$. Thus the elements of $\mathbb{P} \multimap \mathbb{Q}$ are pairs, which we write suggestively as $(p \mapsto q)$, with $p \in \mathbb{P}_\perp, q \in \mathbb{Q}$, ordered by

$$(p' \mapsto q') \leq (p \mapsto q) \iff p \leq p' \ \&\ q' \leq q$$

—note the switch in order on the left.

We have the following chain of isomorphisms between partial orders:

$$\mathbb{P} \otimes \mathbb{Q} \multimap \mathbb{R} = (\mathbb{P} \otimes \mathbb{Q})_\perp \times \mathbb{R} \cong \mathbb{P}_\perp \times \mathbb{Q}_\perp \times \mathbb{R} \cong \mathbb{P} \multimap (\mathbb{Q} \multimap \mathbb{R}) \ .$$

This gives isomorphism between the elements $\mathbb{P} \otimes \widehat{\mathbb{Q} \multimap \mathbb{R}}$ and $\mathbb{P} \multimap \widehat{(\mathbb{Q} \multimap \mathbb{R})}$. Thus there is a 1-1 correspondence *curry* from maps $\mathbb{P} \otimes \mathbb{Q} \to \mathbb{R}$ to maps $\mathbb{P} \to (\mathbb{Q} \multimap \mathbb{R})$ in $\mathcal{D}\mathbf{om}$; its inverse is called *uncurry*. We obtain *linear application*, $app : (\mathbb{P} \multimap \mathbb{Q}) \otimes \mathbb{P} \to \mathbb{Q}$, as $uncurry(1_{\mathbb{P} \multimap \mathbb{Q}})$.

4.3 Products

The product of path orders $\mathbb{P}\&\mathbb{Q}$ is given by the disjoint union of $\mathbb{P}$ and $\mathbb{Q}$. An element of $\widehat{\mathbb{P}\&\mathbb{Q}}$ can be identified with a pair (x, y), with $x \in \widehat{\mathbb{P}}$ and $y \in \widehat{\mathbb{Q}}$, which provides the projections $\pi_1 : \mathbb{P}\&\mathbb{Q} \to \mathbb{P}$ and $\pi_2 : \mathbb{P}\&\mathbb{Q} \to \mathbb{Q}$. More general, not just binary, products $\&_{i \in I} \mathbb{P}_i$ with projections π_j, for $j \in I$, are defined similarly. From the universal

property of products, a collection of maps $f_i : \mathbb{P} \to \mathbb{P}_i$, for $i \in I$, can be tupled together to form a unique map $\langle f_i \rangle_{i \in I} : \mathbb{P} \to \mathbin{\&}_{i \in I} \mathbb{P}_i$ with the property that $\pi_j \circ \langle f_i \rangle_{i \in I} = f_j$ for all $j \in I$. The empty product is given by $\mathbb{O}$ and as the terminal object is associated with unique maps $\mathbb{P} \to \mathbb{O}$, constantly $\emptyset$, for any path order $\mathbb{P}$. Finite products are most often written as $\mathbb{P}_1 \& \cdots \& \mathbb{P}_k$.

Each object $\mathbb{P}$ is associated with (nondeterministic) sum operations, a map $\Sigma : \mathbin{\&}_{i \in I} \mathbb{P} \to \mathbb{P}$ in $\mathcal{D}\mathbf{om}$ taking an element of the domain, viewed as a tuple $\{x_i \mid i \in I\}$, to its union $\bigcup_{i \in I} x_i$ in $\widehat{\mathbb{P}}$. The empty sum yields $\emptyset \in \mathbb{P}$. Finite sums are typically written as $x_1 + \cdots + x_k$.

Because there are empty elements we can define maps in $\mathcal{D}\mathbf{om}_\mathbf{s}$ from products to tensors of path orders. For instance, in the binary case, $\sigma : \mathbb{P} \& \mathbb{Q} \to_s \mathbb{P} \otimes \mathbb{Q}$ in $\mathcal{D}\mathbf{om}_\mathbf{s}$ is specified by

$$(x, y) \mapsto (x \otimes \emptyset) + (\emptyset \otimes y) \ .$$

The composition of such a map with the diagonal map, *viz.*

$$\mathbb{P} \xrightarrow{diag} \mathbb{P} \& \mathbb{P} \xrightarrow{\sigma} \mathbb{P} \otimes \mathbb{P}$$

will play a role later in the semantics of the metalanguage, allowing us to duplicate arguments to maps of a certain kind.

4.4 Lifted sums

The category $\mathcal{D}\mathbf{om}$ does not have coproducts. However, we can build a useful sum in $\mathcal{D}\mathbf{om}$ with the help of the coproduct of $\mathcal{D}\mathbf{om}_\mathbf{s}$ and lifting. Let $\mathbb{P}_i$, for $i \in I$, be a family of path orders. As their lifted sum we take the disjoint union of the path orders $\Sigma_{i \in I} \mathbb{P}_{i\perp}$, over the underlying set $\bigcup_{i \in I} \{i\} \times (\mathbb{P}_i)_\perp$; the latter path order forms a coproduct in $\mathcal{D}\mathbf{om}_\mathbf{s}$ with the obvious injections $in_j : \mathbb{P}_{j\perp} \to_s \Sigma_{i \in I} \mathbb{P}_{i\perp}$, for $j \in I$. The *injections* $In_j : \mathbb{P}_j \to \Sigma_{i \in I} \mathbb{P}_{i\perp}$ in $\mathcal{D}\mathbf{om}$, for $j \in I$, are defined to be the composition $In_j(-) = in_j(\lfloor - \rfloor)$. This construction is not a coproduct in $\mathcal{D}\mathbf{om}$. However, it does satisfy a weaker property analogous to the universal property of a coproduct. Suppose $f_i : \mathbb{P}_i \to \mathbb{Q}$ are maps in $\mathcal{D}\mathbf{om}$ for all $i \in I$. Then, there is a unique mediating map

$$f : \Sigma_{i \in I} \mathbb{P}_{i\perp} \to_s \mathbb{Q}$$

in $\mathcal{D}\mathbf{om}_{\mathbf{s}}$ (note the subscript) such that

$$f \circ In_i = f_i$$

for all $i \in I$.

Suppose that the family of maps $f_i : \mathbb{P}_i \to \mathbb{Q}$, with $i \in I$, has the property that each f_i is constantly $\emptyset$ whenever $i \in I$ is different from j and that f_j is $h : \mathbb{P}_j \to \mathbb{Q}$. Write $[h]_j : \sum_{i \in I} \mathbb{P}_{i\perp} \to \mathbb{Q}$ for the unique mediating map obtained for this choice. Then

$$[h]_j(In_j(z)) = h(z) \ , [h]_j(In_i(z)) = \emptyset \text{ if } i \neq j \ , \text{ and } [h]_j(\emptyset) = \emptyset \ .$$

For a general family $f_i : \mathbb{P}_i \to \mathbb{Q}$, with $i \in I$, we can describe the action of the mediating morphism on $x \in \widehat{\Sigma_{i \in I} \mathbb{P}_{i\perp}}$ as $f(x) = \Sigma_{i \in I}[f_i]_i(x)$.

Because lifted sum is not a coproduct we do not have that tensor distributes over lifted sum to within isomorphism. However there is a map in $\mathcal{D}\mathbf{om}_{\mathbf{s}}$

$$dist : \mathbb{Q} \otimes \Sigma_{i \in I} \mathbb{P}_{i\perp} \to_s \Sigma_{i \in I} (\mathbb{Q} \otimes \mathbb{P}_i)_\perp \ ,$$

expressing a form of distributivity, given as the extension $h^\dagger$ of the function

$$h : \mathbb{Q}_\perp \times (\Sigma_{i \in I} \mathbb{P}_{i\perp})_\perp \to \Sigma_{i \in I} (\mathbb{Q} \otimes \mathbb{P}_i)_\perp; \quad h(q, (i, p)) = (i, (q, p))\downarrow, \ h(q, \perp) = \emptyset \ .$$

Unary lifted sums in $\mathcal{D}\mathbf{om}$, when I is a singleton, are an important special case as they amount to lifting.

4.5 Recursive definitions

Suppose that we wish to model a process language rather like CCS but where processes are passed instead of discrete values, subject to the linearity constraint that when a process is received it can be run at most once. Assume the synchronised communication occurs along channels forming the set A. The path orders can be expected to satisfy the following equations:

$$\mathbb{P} = \mathbb{P}_\perp + \Sigma_{a \in A} \mathbb{C}_\perp + \Sigma_{a \in A} \mathbb{F}_\perp \ , \qquad \mathbb{C} = \mathbb{P} \otimes \mathbb{P} \ , \qquad \mathbb{F} = (\mathbb{P} \multimap \mathbb{P}) \ .$$

The three components of process paths $\mathbb{P}$ represent paths beginning with a silent (τ) action, an output on a channel (a!), resuming as a concretion path (in $\mathbb{C}$), and an input from a channel (a?), resuming as an abstraction path (in $\mathbb{F}$). It is our choice of path for abstractions which narrows us to a *linear* process-passing language, one where the input process can be run at most once to yield a single (computation) path.

Fortunately the simple technique for solving recursive domain equations via information systems in [14] suffices to solve such equations. A path order $\mathbb{P}$ can be regarded as an information system in which every finite subset of $\mathbb{P}$ is consistent and in which the entailment relation is given by the partial order $\leq$ of $\mathbb{P}$, so $\{p'\} \vdash p$ iff $p \leq p'$. Path orders under the order

$$\mathbb{P} \trianglelefteq \mathbb{Q} \iff \mathbb{P} \subseteq \mathbb{Q} \;\&\; (\forall p, p' \in \mathbb{P}.\; p \leq_{\mathbb{P}} p' \iff p \leq_{\mathbb{Q}} p')$$

form a (large) cpo with respect to which all the constructions on path orders we have just seen are continuous (their continuity is verified just as in information systems by showing them monotonic w.r.t. $\trianglelefteq$ and "continuous on token sets"). Solutions to equations like those above are then obtained as (simultaneous) least fixed points.

5 A metalanguage

Assume that path orders are presented using the constructions with the following syntax:

$$\begin{aligned}\mathbb{T} ::= &\mathbb{O} \;\mid\; \mathbb{T}_1 \otimes \mathbb{T}_2 \;\mid\; \mathbb{T}_1 \multimap \mathbb{T}_2 \;\mid\; \Sigma_{i\in I}\mathbb{T}_{i\perp} \;\mid\; \mathbb{T}_1 \& \mathbb{T}_2 \\ &\mid\; P \;\mid\; \mu_j P_1, \cdots, P_k.(\mathbb{T}_1, \cdots, \mathbb{T}_k)\end{aligned}$$

All the construction names have been met earlier with the exception of the notation for recursively defined path orders. Above P is drawn from a set of variables used in the recursive definition of path orders; $\mu_j P_1, \cdots, P_k.(\mathbb{T}_1, \cdots, \mathbb{T}_k)$ stands for the j-component (so $1 \leq j \leq k$) of the $\trianglelefteq$-least solution to the defining equations

$$P_1 = \mathbb{T}_1,\; \cdots,\; P_k = \mathbb{T}_k\;,$$

in which the expressions $\mathbb{T}_1, \cdots, \mathbb{T}_k$ may contain $P_1, \cdots, P_k$. We shall write $\mu P_1, \cdots, P_k.(\mathbb{T}_1, \cdots, \mathbb{T}_k)$ as an abbreviation for

$$(\mu_1 P_1, \cdots, P_k.(\mathbb{T}_1, \cdots, \mathbb{T}_k), \cdots, \mu_k P_1, \cdots, P_k.(\mathbb{T}_1, \cdots, \mathbb{T}_k))\;.$$

In future we will often use vector notation and, for example, write $\mu\overrightarrow{P}.\overrightarrow{\mathbb{T}}$ for the expression above, and confuse a closed expression for a path order with the path order itself.

The operations of Sections 3 and 4 form the basis of a "raw" syntax of terms which will be subject to typing and linearity constraints later:

$$
\begin{array}{lll}
t, u, v, \cdots ::= & x, y, z, \cdots & \text{(Variables)} \\
& \emptyset \ \mid \ \Sigma_{i\in I} t_i \ \mid & \text{(Sums)} \\
& \mathit{rec}\ x.t \ \mid & \text{(Recursive definitions)} \\
& \lambda x.t \ \mid \ u \cdot v \ \mid & \text{(Abstraction and application)} \\
& \mathit{In}_j(t) \ \mid \ [t > \mathit{In}_j(x) \Rightarrow u] \ \mid & \text{(Injections and tests for lifted sums)} \\
& (t, u) \ \mid \ [t > (x, -) \Rightarrow u] \ \mid & \\
& \qquad [t > (-, x) \Rightarrow u] \ \mid & \text{(Pairing and tests for products)} \\
& t \otimes u \ \mid \ [t > x \otimes y \Rightarrow u] & \text{(Tensor operation and tests)}
\end{array}
$$

The language is similar to that in [1], being based on a form of pattern matching. In particular $[t > \mathit{In}_j(x) \Rightarrow u]$ "tests"or matches t denoting an element of a lifted sum against the pattern $\mathit{In}_j(x)$ and passes the results of successful matches for x on to u; how the possibly multiple results of successful matches are combined to a final result varies according to the category in which language is interpreted. Accordingly, variables like x in such patterns are binding occurrences and bind later occurrences of the variable in the body, u in this case. We shall take for granted an understanding of free and bound variables, and substitution on raw terms. In examples we'll allow ourselves to use $+$ both in writing sums of terms and lifted sums of path orders.

Let $\mathbb{P}_1, \cdots, \mathbb{P}_k$ be closed expressions for path orders and assume that the variables $x_1, \cdots, x_k$ are distinct. A syntactic judgement

$$x_1 : \mathbb{P}_1, \cdots, x_k : \mathbb{P}_k \vdash t : \mathbb{Q}$$

stands for a map

$$[\![x_1 : \mathbb{P}_1, \cdots, x_k : \mathbb{P}_k \vdash t : \mathbb{Q}]\!] : \mathbb{P}_1 \otimes \cdots \otimes \mathbb{P}_k \to \mathbb{Q}$$

in $\mathcal{D}\mathbf{om}$. We shall typically write Γ, or Δ, for an environment list $x_1 : \mathbb{P}_1, \cdots, x_k : \mathbb{P}_k$. We shall most often abbreviate the denotation map to

$$\mathbb{P}_1 \otimes \cdots \otimes \mathbb{P}_k \xrightarrow{t} \mathbb{Q} \ , \ \text{or even} \ \Gamma \xrightarrow{t} \mathbb{Q} \ .$$

Here k may be 0 so the list in the syntactic judgement is empty and the corresponding tensor product the empty path order $\mathbb{O}$.

A linear language will restrict copying and so substitutions of a common term into distinct variables. The counterpart in the models is the absence of a suitable diagonal map from objects $\mathbb{P}$ to $\mathbb{P} \otimes \mathbb{P}$. For example the function $x \mapsto x \otimes x$ from $\widehat{\mathbb{P}}$ to $\widehat{\mathbb{P} \otimes \mathbb{P}}$ is not in general a map in $\mathcal{D}\mathbf{om}$. To see this assume that $\mathbb{P}$ is the discrete order on the set $\{a, b\}$. Then the nonempty join $x = a{\downarrow} \cup b{\downarrow}$ is not sent to

$$(a{\downarrow} \otimes a{\downarrow}) \cup (b{\downarrow} \otimes b{\downarrow}) = \{(a, a), (b, b), (a, \bot), (\bot, b)\}$$

as would be needed to preserve non-empty joins, but instead to

$$x \otimes x = \{(a, a), (b, b), (a, b), (a, \bot), (\bot, b)\}$$

with the extra "cross term" (a, b). Consider a term $t(x, y)$, with its free variables x and y shown explicitly, for which

$$x : \mathbb{P}, y : \mathbb{P} \vdash t(x, y) : \mathbb{Q} \ ,$$

corresponding to a map $\mathbb{P} \otimes \mathbb{P} \xrightarrow{t(x,y)} \mathbb{Q}$ in $\mathcal{D}\mathbf{om}$. This does not generally entail that

$$x : \mathbb{P} \vdash t(x, x) : \mathbb{Q}$$

—there may not be a corresponding map in $\mathcal{D}\mathbf{om}$, for example if $t(x, y) = x \otimes y$. There is however a condition on how the variables x and y occur in t which ensures that the judgement $x : \mathbb{P} \vdash t(x, x) : \mathbb{Q}$ holds and that it denotes the map in $\mathcal{D}\mathbf{om}$ obtained as the composition

$$\mathbb{P} \xrightarrow{diag} \mathbb{P}\&\mathbb{P} \xrightarrow{\sigma} \mathbb{P} \otimes \mathbb{P} \xrightarrow{t(x,y)} \mathbb{Q}$$

—using the maps seen earlier in Section 4.3. Semantically, the map $\mathbb{P} \otimes \mathbb{P} \xrightarrow{t(x,y)} \mathbb{Q}$ has to be essentially a map $\mathbb{P}\&\mathbb{P} \to \mathbb{Q}$, more precisely the left Kan extension of such a map along σ. Syntactically, this is assured if the variables x and y are *not crossed* in t according to the following definition:

Definition 2. Let t be a raw term. Say a set of variables V is *crossed* in t iff there are subterms of t of the form

a tensor $s \otimes u$, an application $s \cdot u$, or a test $[z > u \Rightarrow s]$

for which t has free occurrences of variables from V appearing in both s and u.

For example, variables x and y are crossed in $x \otimes y$, but variables x and y are not crossed in $(x + y) \otimes z$. Note that a set of variables V is crossed in a term t if V contains variables x, y, not necessarily distinct, so that $\{x, y\}$ is crossed in t. We are mainly interested in when sets of variables are *not crossed* in a term.

The term-formation rules are listed below alongside their interpretations as a constructors on morphisms, taking the morphisms denoted by the premises to that denoted by the conclusion (along the lines of [2]). We assume that the variables in any enviroment list which appears are distinct.

Structural rules:

$$\frac{}{x : \mathbb{P} \vdash x : \mathbb{P}} \text{ , interpreted as } \frac{}{\mathbb{P} \xrightarrow{1_{\mathbb{P}}} \mathbb{P}} .$$

$$\frac{\Delta \vdash t : \mathbb{P}}{\Gamma, \Delta \vdash t : \mathbb{P}} \text{ , interpreted as } \frac{\Delta \xrightarrow{t} \mathbb{P}}{\Gamma \otimes \Delta \xrightarrow{\emptyset \otimes 1_\Gamma} \mathbb{O} \otimes \Delta \cong \Delta \xrightarrow{t} \mathbb{P}} .$$

$$\frac{\Gamma, x : \mathbb{P}, y : \mathbb{Q}, \Delta \vdash t : \mathbb{R}}{\Gamma, y : \mathbb{Q}, x : \mathbb{P}, \Delta \vdash t : \mathbb{R}} \text{ , interpreted via } s : \mathbb{Q} \otimes \mathbb{P} \cong \mathbb{P} \otimes \mathbb{Q} \text{ as}$$

$$\frac{\Gamma \otimes \mathbb{P} \otimes \mathbb{Q} \otimes \Delta \xrightarrow{t} \mathbb{R}}{\Gamma \otimes \mathbb{Q} \otimes \mathbb{P} \otimes \Delta \xrightarrow{1_\Gamma \otimes s \otimes 1_\Delta} \Gamma \otimes \mathbb{P} \otimes \mathbb{Q} \otimes \Delta \xrightarrow{t} \mathbb{R}} .$$

Recursive path orders:

$$\frac{\Gamma \vdash t : \mathbb{T}_j[\mu \overrightarrow{P}.\overrightarrow{\mathbb{T}}/\overrightarrow{P}]}{\Gamma \vdash t : \mu_j \overrightarrow{P}.\overrightarrow{\mathbb{T}}} , \qquad \frac{\Gamma \vdash t : \mu_j \overrightarrow{P}.\overrightarrow{\mathbb{T}}}{\Gamma \vdash t : \mathbb{T}_j[\mu \overrightarrow{P}.\overrightarrow{\mathbb{T}}/\overrightarrow{P}]} .$$

where the premise and conclusion of each rule are interpreted as the same map because $\mu_j \overrightarrow{P}.\overrightarrow{\mathbb{T}}$ and $\mathbb{T}_j[\mu \overrightarrow{P}.\overrightarrow{\mathbb{T}}/\overrightarrow{P}]$ denote equal path orders.

Sums of terms:

$$\frac{}{\Gamma \vdash \emptyset : \mathbb{P}} \text{ , interpreted as } \frac{}{\Gamma \xrightarrow{\emptyset} \mathbb{P}} \text{ , the constantly } \emptyset \text{ map.}$$

$$\frac{\Gamma \vdash t_i : \mathbb{P} \text{ for all } i \in I}{\Gamma \vdash \Sigma_{i \in I} t_i : \mathbb{P}} \text{ , interpreted as } \frac{\Gamma \xrightarrow{t_i} \mathbb{P} \text{ for all } i \in I}{\Gamma \xrightarrow{\langle t_i \rangle_{i \in I}} \&_{i \in I} \mathbb{P} \xrightarrow{\Sigma} \mathbb{P}} .$$

Recursive definitions:

$$\frac{\Gamma, x:\mathbb{P} \vdash t:\mathbb{P} \quad \{y,x\} \text{ not crossed for all } y \text{ in } \Gamma}{\Gamma \vdash rec\ x.t : \mathbb{P}} \text{ , interpreted as } \frac{\Gamma \otimes \mathbb{P} \xrightarrow{t} \mathbb{P}}{\Gamma \xrightarrow{fix\ F} \mathbb{P}}$$

—see Section 3.4, where for $\Gamma \xrightarrow{g} \mathbb{P}$ the map $F(g)$ is the composition

$$\Gamma \xrightarrow{diag} \Gamma \& \Gamma \xrightarrow{\sigma} \Gamma \otimes \Gamma \xrightarrow{1_\Gamma \otimes g} \Gamma \otimes \mathbb{P} \xrightarrow{t} \mathbb{P} \ .$$

Abstraction:

$$\frac{\Gamma, x:\mathbb{P} \vdash t:\mathbb{Q}}{\Gamma \vdash \lambda x.t : \mathbb{P} \multimap \mathbb{Q}} \text{ , interpreted as } \frac{\Gamma \otimes \mathbb{P} \xrightarrow{t} \mathbb{Q}}{\Gamma \xrightarrow{curry\ t} (\mathbb{P} \multimap \mathbb{Q})} \ .$$

Application:

$$\frac{\Gamma \vdash u : \mathbb{P} \multimap \mathbb{Q} \ \Delta \vdash v : \mathbb{P}}{\Gamma, \Delta \vdash u \cdot v : \mathbb{Q}} \text{ ,interpreted as } \frac{\Gamma \xrightarrow{u} (\mathbb{P} \multimap \mathbb{Q}) \quad \Delta \xrightarrow{v} \mathbb{P}}{\Gamma \otimes \Delta \xrightarrow{u \otimes v} (\mathbb{P} \multimap \mathbb{Q}) \otimes \mathbb{P} \xrightarrow{app} \mathbb{Q}} \ .$$

Injections and test for lifted sums:

$$\frac{\Gamma \vdash t : \mathbb{P}_j \ , \text{ where } j \in I}{\Gamma \vdash In_j(t) : \Sigma_{i \in I} \mathbb{P}_{i\perp}} \text{ , interpreted as } \frac{\Gamma \xrightarrow{t} \mathbb{P}_j \ , \text{ where } j \in I}{\Gamma \xrightarrow{t} \mathbb{P}_j \xrightarrow{In_j} \Sigma_{i \in I} \mathbb{P}_{i\perp}} \ .$$

$$\frac{\Gamma, x : \mathbb{P}_j \vdash u : \mathbb{Q} \ , \text{ where } j \in I. \quad \Delta \vdash t : \sum_{i \in I} \mathbb{P}_{i\perp}}{\Gamma, \Delta \vdash [t > In_j(x) \Rightarrow u] : \mathbb{Q}} \text{ , interpreted as}$$

$$\frac{\Gamma \otimes \mathbb{P}_j \xrightarrow{u} \mathbb{Q} \text{ , where } j \in I. \quad \Delta \xrightarrow{t} \sum_{i \in I} \mathbb{P}_{i\perp}}{\Gamma \otimes \Delta \xrightarrow{1_\Gamma \otimes t} \Gamma \otimes \sum_{i \in I} \mathbb{P}_{i\perp} \xrightarrow{dist} \sum_{i \in I} (\Gamma \otimes \mathbb{P}_i)_\perp \xrightarrow{[-]_j} \mathbb{Q}} \ .$$

Pairing and tests for products:

$$\frac{\Gamma \vdash t : \mathbb{P} \quad \Gamma \vdash u : \mathbb{Q}}{\Gamma \vdash (t,u) : \mathbb{P} \& \mathbb{Q}} \text{ , interpreted as } \frac{\Gamma \xrightarrow{t} \mathbb{P} \quad \Gamma \xrightarrow{u} \mathbb{Q}}{\Gamma \xrightarrow{\langle t,u \rangle} \mathbb{P} \& \mathbb{Q}} \ .$$

$$\frac{\Gamma, x : \mathbb{P} \vdash u : \mathbb{R} \quad \Delta \vdash t : \mathbb{P} \& \mathbb{Q}}{\Gamma, \Delta \vdash [t > (x, -) \Rightarrow u] : \mathbb{R}}, \text{ interpreted as}$$

$$\frac{\Gamma \otimes \mathbb{P} \xrightarrow{u} \mathbb{R} \quad \Delta \xrightarrow{t} \mathbb{P} \& \mathbb{Q}}{\Gamma \otimes \Delta \xrightarrow{1_\Gamma \otimes (\pi_1 \circ t)} \Gamma \otimes \mathbb{P} \xrightarrow{u} \mathbb{R}} \ .$$

$\dfrac{\Gamma, x : \mathbb{Q} \vdash u : \mathbb{R} \quad \Delta \vdash t : \mathbb{P}\&\mathbb{Q}}{\Gamma, \Delta \vdash [t > (-, x) \Rightarrow u] : \mathbb{R}}$, interpreted as

$$\frac{\Gamma \otimes \mathbb{Q} \xrightarrow{u} \mathbb{R} \quad \Delta \xrightarrow{t} \mathbb{P}\&\mathbb{Q}}{\Gamma \otimes \Delta \xrightarrow{1_\Gamma \otimes (\pi_2 \circ t)} \Gamma \otimes \mathbb{Q} \xrightarrow{u} \mathbb{R}} .$$

Tensor operation and test for tensor:

$$\frac{\Gamma \vdash t : \mathbb{P} \quad \Delta \vdash u : \mathbb{Q}}{\Gamma, \Delta \vdash t \otimes u : \mathbb{P} \otimes \mathbb{Q}} \text{, interpreted as } \frac{\Gamma \xrightarrow{t} \mathbb{P} \quad \Delta \xrightarrow{u} \mathbb{Q}}{\Gamma \otimes \Delta \xrightarrow{t \otimes u} \mathbb{P} \otimes \mathbb{Q}} .$$

$\dfrac{\Gamma, x : \mathbb{P}, y : \mathbb{Q} \vdash u : \mathbb{R} \quad \Delta \vdash t : \mathbb{P} \otimes \mathbb{Q}}{\Gamma, \Delta \vdash [t > x \otimes y \Rightarrow u] : \mathbb{R}}$, interpreted as

$$\frac{\Gamma \otimes \mathbb{P} \otimes \mathbb{Q} \xrightarrow{u} \mathbb{R} \quad \Delta \xrightarrow{t} \mathbb{P} \otimes \mathbb{Q}}{\Gamma \otimes \Delta \xrightarrow{1_\Gamma \otimes t} \Gamma \otimes \mathbb{P} \otimes \mathbb{Q} \xrightarrow{u} \mathbb{R}} .$$

Proposition 3. *Suppose $\Gamma, x : \mathbb{P} \vdash t : \mathbb{Q}$. The set $\{x\}$ is not crossed in t.*

Lemma 4. *(Well-formed substitutions) Suppose*

$$\Gamma, x_1 : \mathbb{P}, \cdots, x_k : \mathbb{P} \vdash t : \mathbb{Q}$$

and that the set of variables $\{x_1, \cdots, x_k\}$ is not crossed in t. Suppose $\Delta \vdash u : \mathbb{P}$ where the variables of Γ and Δ are disjoint. Then,

$$\Gamma, \Delta \vdash t[u/x_1, \cdots, u/x_k] : \mathbb{Q} .$$

In particular, as singleton sets of variables are not crossed in well-formed terms we immediately deduce:

Corollary 5. *If $\Gamma, x : \mathbb{P} \vdash t : \mathbb{Q}$ and $\Delta \vdash u : \mathbb{P}$, where the variables of Γ and Δ are disjoint, then $\Gamma, \Delta \vdash t[u/x] : \mathbb{Q}$.*

Exploiting the naturality of the various operations used in the semantic definitions, we can show:

Lemma 6. *(Substitution Lemma) Suppose $\Gamma, x : \mathbb{P} \vdash t : \mathbb{Q}$ and $\Delta \vdash u : \mathbb{P}$ where Γ and Δ have disjoint variables. Then,*

$$[\![\Gamma, \Delta \vdash t[u/x] : \mathbb{Q}]\!] = [\![\Gamma, x : \mathbb{P} \vdash t : \mathbb{Q}]\!] \circ (1_\Gamma \otimes [\![\Delta \vdash u : \mathbb{P}]\!]) .$$

In particular, linear application amounts to substitution:

Lemma 7. *Suppose $\Gamma \vdash (\lambda x.t) \cdot u : \mathbb{Q}$. Then, $\Gamma \vdash t[u/x] : \mathbb{Q}$ and*

$$[\![\Gamma \vdash (\lambda x.t) \cdot u : \mathbb{Q}]\!] = [\![\Gamma \vdash t[u/x] : \mathbb{Q}]\!] .$$

5.1 Extending the metalanguage

General patterns are well-formed terms built up according to

$$p ::= x \mid \emptyset \mid In_j(p) \mid p \otimes q \mid (p, -) \mid (-, p) \mid p \mapsto p' \ .$$

A test on a pattern $[u > p \Rightarrow t]$ binds the free variables of the pattern p to the resumptions after following the path specified by the pattern in u; because the term t may contain these variables freely the resumptions may influence the computation of t. Such a test is understood inductively as an abbreviation for a term in the metalanguage:

$$\begin{array}{ll}
[u > x \Rightarrow t] \equiv (\lambda x.t) \cdot u \ , & [u > \emptyset \Rightarrow t] \equiv t \ , \\
[u > In_j(p) \Rightarrow t] \equiv [u > In_j(x) \Rightarrow [x > p \Rightarrow t]] & \text{for a fresh variable } x, \\
[u > (p, -) \Rightarrow t] \equiv [u > (x, -) \Rightarrow [x > p \Rightarrow t]] & \text{for a fresh variable } x, \\
[u > (-, p) \Rightarrow t] \equiv [u > (-, x) \Rightarrow [x > p \Rightarrow t]] & \text{for a fresh variable } x, \\
[u > p \otimes q \Rightarrow t] \equiv [u > x \otimes y \Rightarrow [p > x \Rightarrow [q > y \Rightarrow t]]] & \text{for fresh variables } x, y, \\
[u > (p \mapsto q) \Rightarrow t] \equiv [u > f \Rightarrow [f \cdot p > q \Rightarrow t]] & \text{for a fresh variable } f.
\end{array}$$

Let $\lambda x \otimes y.t$ stand for $\lambda w.[w > x \otimes y \Rightarrow t]$, where w is a fresh variable, and write $[u_1 > p_1, \cdots, u_k > p_k \Rightarrow t]$ to abbreviate $[u_1 > p_1 \Rightarrow [\cdots [u_k > p_k \Rightarrow t] \cdots]$.

5.2 Interpretation in $\mathcal{C}$**on**

We can interpret the metalanguage in the category of presheaf models $\mathcal{C}$**on** with essentially the same constructions and operations as those in $\mathcal{D}$**om**, once we replace **2** by **Set** and understand (nonempty) joins as (connected) colimits; [3] the category of presheaf models $\mathcal{C}$**ol** will play the role of $\mathcal{D}\mathbf{om}_\mathbf{s}$. Because now domains $[\mathbb{P}^{op}, \mathbf{2}]$ are replaced by presheaf categories $[\mathbb{P}^{op}, \mathbf{Set}]$ we shall often have to make do with isomorphism rather than straight equality.

In fact, to mimic the mathematics behind the interpretation of the metalanguage in $\mathcal{D}$**om**, all that's required of a category $\mathcal{V}$, in place of **2**, is that it has all *colimits* and all *finite products*. Now $\widehat{\mathbb{P}}$,

[3] A function between partial orders with least elements preserves (connected) colimits iff it preserves (nonempty) joins.

taken to be $[\mathbb{P}^{op}, \mathcal{V}]$, will have all colimits, in particular coproducts to interpret (nondeterministic) sums, and will also support left Kan extensions to play the role of $(-)^\dagger$. We can understand the embedding $(-)\downarrow : \mathbb{P} \to \widehat{\mathbb{P}}$ through the initial and terminal objects $\mathbf{0}$ and $\mathbf{1}$ of $\mathcal{V}$. The lifting map $\lfloor - \rfloor : \mathbb{P} \to \widehat{\mathbb{P}}$, again defined so $\lfloor x \rfloor(\bot) = \mathbf{1}$ and $\lfloor x \rfloor(p\downarrow) = x(p)$, will preserve connected colimits. Using the product of $\mathcal{V}$, instead of that of $\mathbf{2}$, we can copy the definition of the functor $\otimes$.

The advantage of this generality is that objects in the category $\mathcal{V}$ don't just have to say whether a path is present in a process but can provide a "measure" of how. If $\mathcal{V}$ is **Set** a process will denote a presheaf X which identifies the set of different ways $X(p)$ in which a path p is realised.

6 Examples

6.1 CCS

As in CCS, assume a set of labels A, a complementation operation producing $\bar{a}$ from a label a, with $\bar{\bar{a}} = a$, and a distinct label τ. In the metalanguage we can specify the path order $\mathbb{P}$ as the $\trianglelefteq$-least solution

$$\mathbb{P} = \mathbb{P}_\bot + \Sigma_{a \in A}\mathbb{P}_\bot + \Sigma_{a \in A}\mathbb{P}_\bot \ .$$

Write the injections from $\mathbb{P}$ into its expression as a lifted sum as $\tau.t$, $a.t$ and $\bar{a}.t$ for $a \in A$ and term t of type $\mathbb{P}$. The curried CCS parallel composition can be defined as the following term of type $\mathbb{P} \multimap (\mathbb{P} \multimap \mathbb{P})$ in the metalanguage:

$$\begin{aligned} Par = rec\ P.\ \lambda x \lambda y.\ & \Sigma_{\alpha \in A \cup \{\tau\}}[x > \alpha.x' \Rightarrow \alpha.(P \cdot x' \cdot y)] + \\ & \Sigma_{\alpha \in A \cup \{\tau\}}[y > \alpha.y' \Rightarrow \alpha.(P \cdot x \cdot y')] + \\ & \Sigma_{a \in A}[x > a.x',\ y > \bar{a}.y' \Rightarrow \tau.(P \cdot x' \cdot y')] \ . \end{aligned}$$

The other CCS operations are easy to encode. Interpreted in $\mathcal{D}$**om** two CCS terms will have the same denotation iff they have same traces (or execution sequences). By virtue of having been written down in the metalanguage the operation of parallel composition will preserve open-map bisimulation when interpreted in $\mathcal{C}$**on**; for this specific $\mathbb{P}$, open-map bisimulation coincides with strong bisimulation.

In $\mathcal{C}\mathbf{on}$ we can recover the expansion law for general reasons: the Substitution Lemmas 6,7 hold in $\mathcal{C}\mathbf{on}$, though with isomorphism replacing equality; the mediating morphism associated with lifted sums are now in $\mathcal{C}\mathbf{ol}$ (the analogue of $\mathcal{D}\mathbf{om}_\mathbf{s}$) so that tests for lifted sums distribute over nondeterministic sums. In more detail, write $X|Y$ for $Par \cdot X \cdot Y$, where X and Y are terms of type $\mathbb{P}$. Suppose

$$X = \Sigma_{\alpha \in A \cup \{\tau\}} \Sigma_{i \in I(\alpha)} \alpha.X_i \ , \qquad Y = \Sigma_{\alpha \in A \cup \{\tau\}} \Sigma_{j \in J(\alpha)} \alpha.Y_j \ .$$

Using Lemma 7, and then that the tests distribute over nondeterministic sums,

$$\begin{aligned} X|Y \cong & \Sigma_{\alpha \in A \cup \{\tau\}} [X > \alpha.x' \Rightarrow \alpha.(x'|Y)] + \Sigma_{\alpha \in A \cup \{\tau\}} [Y > \alpha.y' \Rightarrow \alpha.(X|y')] \\ & + \Sigma_{a \in A} [X > a.x', \ Y > \bar{a}.y' \Rightarrow \tau.(x'|y')] \\ \cong & \Sigma_{\alpha \in A \cup \{\tau\}} \Sigma_{i \in I(\alpha)} \alpha.(X_i|Y) + \Sigma_{\alpha \in A \cup \{\tau\}} \Sigma_{j \in J(\alpha)} \alpha.(X|Y_j) \\ & + \Sigma_{a \in A} \Sigma_{i \in I(a), j \in J(\bar{a})} \tau.(X_i|Y_j) \ . \end{aligned}$$

The equation for the path order for CCS with *early value-passing* would be very similar to that above. An equation suitable for *late value-passing* is

$$\mathbb{P} = \mathbb{P}_\perp + \Sigma_{a \in A, v \in V} \mathbb{P}_\perp + \Sigma_{a \in A} (\Sigma_{v \in V} \mathbb{P}_\perp)_\perp \ ,$$

though this is not the same equation as in [20] which has $\Sigma_{a \in A} (\Sigma_{v \in V} \mathbb{P})_\perp$ as the final component—perhaps the metalanguage should be broadened to allow this.

6.2 A linear higher-order process language

Recall the path orders for processes, concretions and abstractions for a higher-order language in Section 4.5. We are chiefly interested in the parallel composition of processes, $Par_{\mathbb{P},\mathbb{P}}$ of type $\mathbb{P} \otimes \mathbb{P} \multimap \mathbb{P}$. But parallel composition is really a family of mutually dependent operations also including components such as $Par_{\mathbb{F},\mathbb{C}}$ of type $\mathbb{F} \otimes \mathbb{C} \multimap \mathbb{F}$ to say how abstractions compose in parallel with concretions *etc.* All these components can be tupled together in a product using &, and parallel composition defined as a simultaneous recursive definition

whose component at $\mathbb{P} \otimes \mathbb{P} \multimap \mathbb{P}$ satisfies

$$\begin{aligned} P|Q = & \Sigma_\alpha[P > \alpha.P' \Rightarrow \alpha(P'|Q)] + \\ & \Sigma_\alpha[Q > \alpha.Q' \Rightarrow \alpha(P|Q')] + \\ & \Sigma_a[P > a?F,\ Q > a!S \otimes R \Rightarrow \tau.(F \cdot S|R)] + \\ & \Sigma_a[P > a!S \otimes R,\ Q > a?F \Rightarrow \tau.(R|F \cdot S)] \ , \end{aligned}$$

where we have chosen suggestive names for the injections and, for instance, $P|Q$ abbreviates $Par_{\mathbb{P},\mathbb{P}} \cdot (P \otimes Q)$. In the summations $a \in A$ and α ranges over $a!, a?, \tau$ for $a \in A$.

7 Problems

The interpretation of the metalanguage in $\mathcal{C}\mathbf{on}$ provides a base from which to examine its equational theory and operational semantics. We should update the treatment of bisimulation in [4] to take better account of $\mathcal{C}\mathbf{on}$ and the metalanguage.

The range of interpretations for the metalanguage indicated in Section 5.2 is restrictive, for example, in requiring $\mathcal{V}$ to be cocomplete. As remarked in [8] there are sensible choices for $\mathcal{V}$ which are not cocomplete—the countable sets for instance, provided we also restrict the path orders to be countable.

Perhaps instantiating $\mathcal{V}$ to some specific category, can help provide a "presheaf model" of a higher-order Pi-Calculus to accompany [5]. This would be a good basis from which to compare and relate with the project of action structures [16].

The metalanguage here cries out for extensions in two directions, one to cope with name generation as in the Pi-Calculus, the other to go beyond linearity. The exponential ! of [20, 5] seems appropriate but its effects on open-map bisimulation are not understood.

The question of how to approach higher-order independence models remains.

How to turn the framework on weak bisimulation and contextual equivalence is the subject of current work based on a lead by Marcelo Fiore.

Gian Luca Cattani and I are working on how to understand open-map bisimulation at higher-order in operational terms [7].

References

1. S. Abramsky. Computational interpretation of linear logic. Tech. Report 90/20, Dept. of Computing, Imperial College, 1990.
2. T. Braüner. An Axiomatic Approach to Adequacy. BRICS Dissertation Series DS-96-4, 1996.
3. G. L. Cattani. Forthcoming PhD thesis, CS Dept., University of Aarhus.
4. G. L. Cattani, M. Fiore, and G. Winskel. A Theory of Recursive Domains with Applications to Concurrency. In *Proc. of LICS '98*.
5. G. L. Cattani, I. Stark, and G. Winskel. Presheaf Models for the π-Calculus. In *Proc. of CTCS '97*, LNCS 1290, 1997.
6. G. L. Cattani and G. Winskel. Presheaf Models for Concurrency. In *Proc. of CSL' 96*, LNCS 1258, 1997.
7. G. L. Cattani and G. Winskel. On bisimulation for higher order processes. Manuscript, 1998.
8. G. L. Cattani, A. J. Power and G. Winskel. A categorical axiomatics for bisimulation. In *Proc. of CONCUR'98*, LNCS 1466, 1998.
9. M. Hennessy. A fully abstract denotational semantics for the pi-calculus Computer Science Technical Report 96:04, Sussex University, 1996.
10. M. Hennessy and G.D. Plotkin. Full abstraction for a simple parallel programming language. In *Proc. of MFCS'79*, LNCS 74, 1979.
11. C.A.R. Hoare. A model for communicating sequential processes. *Tech. Report PRG-22, University of Oxford Computing Lab.*, 1981.
12. A. Joyal and I. Moerdijk. A completeness theorem for open maps. *Annals of Pure and Applied Logic*, 70:51–86, 1994.
13. A. Joyal, M. Nielsen, and G. Winskel. Bisimulation from open maps. *Information and Computation*, 127:164–185, 1996.
14. G.Winskel and K.Larsen. Using information systems to solve recursive domain equations effectively. LNCS 173, 1984.
15. R. Milner. *Communication and concurrency*. Prentice Hall, 1989.
16. R. Milner. Calculi for Interaction. *Acta Informatica* 33, 1996.
17. M. Nielsen, G.D. Plotkin and G. Winskel. Petri nets, Event structures and Domains, part 1. Theoretical Computer Science, vol. 13, 1981.
18. G. Winskel. A representation of completely distributive algebraic lattices. Report of the Computer Science Dept., Carnegie-Mellon University, 1983.
19. G. Winskel. An introduction to event structures. In *Proc. of REX summerschool in temporal logic, 'May 88*, LNCS 354, 1988.
20. G. Winskel. A presheaf semantics of value-passing processes. In *Proceedings of CONCUR'96*, LNCS 1119, 1996.

Recent BRICS Report Series Publications

RS-98-31 **Glynn Winskel.** ***A Linear Metalanguage for Concurrency*****. November 1998. 21 pp.**

RS-98-30 **Carsten Butz.** ***Finitely Presented Heyting Algebras*****. November 1998. 30 pp.**

RS-98-29 **Jan Camenisch and Markus Michels.** ***Proving in Zero-Knowledge that a Number is the Product of Two Safe Primes*****. November 1998. 19 pp.**

RS-98-28 **Rasmus Pagh.** ***Low Redundancy in Dictionaries with*** $O(1)$ ***Worst Case Lookup Time*****. November 1998. 15 pp.**

RS-98-27 **Jan Camenisch and Markus Michels.** ***A Group Signature Scheme Based on an RSA-Variant.*** **November 1998. 18 pp. Preliminary version appeared in Ohta and Pei, editors,** ***Advances in Cryptology: 4th ASIACRYPT Conference on the Theory and Applications of Cryptologic Techniques,*** **ASIACRYPT '98 Proceedings, LNCS 1514, 1998, pages 160–174.**

RS-98-26 **Paola Quaglia and David Walker.** ***On Encoding*** $p\pi$ ***in*** $m\pi$**. October 1998. 27 pp. Full version of paper to appear in** ***Foundations of Software Technology and Theoretical Computer Science: 18th Conference,*** **FCT&TCS '98 Proceedings, LNCS, 1998.**

RS-98-25 **Devdatt P. Dubhashi.** ***Talagrand's Inequality in Hereditary Settings*****. October 1998. 22 pp.**

RS-98-24 **Devdatt P. Dubhashi.** ***Talagrand's Inequality and Locality in Distributed Computing*****. October 1998. 14 pp.**

RS-98-23 **Devdatt P. Dubhashi.** ***Martingales and Locality in Distributed Computing*****. October 1998. 19 pp.**

RS-98-22 **Gian Luca Cattani, John Power, and Glynn Winskel.** ***A Categorical Axiomatics for Bisimulation*****. September 1998. ii+21 pp. Appears in Sangiorgi and de Simone, editors,** ***Concurrency Theory: 9th International Conference,*** **CONCUR '98 Proceedings, LNCS 1466, 1998, pages 581–596.**

RS-98-21 **John Power, Gian Luca Cattani, and Glynn Winskel.** ***A Representation Result for Free Cocompletions*****. September 1998. 16 pp.**