An Order-Theoretic Model for the Algebra of Communicating Processes

P.M.W. Knijnenburg

RUU-CS-92-02
January 1992

Utrecht University

Department of Computer Science

Padualaan 14, P.O. Box 80.089,
3508 TB Utrecht, The Netherlands,
Tel. : ... + 31 - 30 - 531454

An Order-Theoretic Model for the Algebra of Communicating Processes

P.M.W. Knijnenburg

Technical Report RUU-CS-92-02
January 1992

Department of Computer Science
Utrecht University
P.O.Box 80.089
3508 TB Utrecht
The Netherlands

ISSN: 0924–3275

An Order-Theoretic Model for the Algebra of Communicating Processes

P.M.W. Knijnenburg
Utrecht University *

Abstract

In this paper an order-theoretic interpretation for the process algebra *ACP* is developed. We show that it is a model for the theory by showing that the axioms for *ACP* hold in the interpretation. We furthermore show that the model is complete in the sense that two *ACP* terms are equal if and only if they are equal in the model.

1 Introduction

In the nineteenseventies a new and fruitful branch of computer science emerged, namely the theory of processes. In the algebraic version of it, the immense complexities of reasoning about the behavior of concurrent systems were reduced by identifying small sets of 'primitive' combinators and small collections of basic equalities that the terms of the algebra, *c.q.* the processes, should obey. Examples include Milner's *CCS* [Mil80], Hoare's *CSP* [Hoa85] and Bergstra and Klop's *ACP* [BK84, BK85]. In this paper we focus attention to the latter.

Traditionally, *ACP* is interpreted over a domain of finitely branching process graphs modulo bisimulation. It is known that an order theoretic model for *ACP* is possible, but such a model has, to our knowledge, never been built. This paper closes the gap by defining in detail an interpretation (*i.e.* a denotational semantics) for the algebra, which has an ω-algebraic complete partial order, $\mathbf{P}$, as its target. The main result of the paper is a proof that the interpretation is a sound and complete model. That is, every equality between terms of the algebra *ACP* holds if and only if it holds in the model.

$\mathbf{P}$ itself is defined by means of a reflexive domain equation: $\mathbf{P} \cong \mathcal{P}^*(\{\delta\}_\perp + A_\perp + A_\perp \times \mathbf{P})$. Thus defined, $\mathbf{P}$ is too large: it also contains elements that are not the denotation of any term. We identify an inclusive subset $\mathbf{P}' \subset \mathbf{P}$ such that the denotation of a term always lies in $\mathbf{P}'$ and $\mathbf{P}'$ is the smallest such subset. We need the whole of $\mathbf{P}$, however, to define our semantical operators: intermediate values may fall outside $\mathbf{P}'$.

The main tool we employ in defining our semantical operators is a typed lambda calculus. The category **Cpo** of complete partially ordered sets with continuous maps is

*Dept. of Computer Science, Padualaan 14, P.O.Box 80.089, 3508 TB Utrecht, The Netherlands. E-mail: peterk@cs.ruu.nl

cartesian closed [LS86]. Hence it has a typed lambda calculus as its internal language. Any continuous function occurs as a constant in this language. Furthermore, any typed lambda term in the language has an interpretation as a continuous function (arrow) in the category. This implies that we only need to identify a small set of primitive continuous functions in order to be able to define complex functions using the typed lambda calculus. These complex functions are then *by definition* continuous. This approach can be summarized by saying that 'careful informal reasoning induces precise formal definitions'. Apart from the main result, we feel that another contribution of the paper consists of the clear and concise definition of a denotational semantics to a non-trivial, reflexive domain. Thus the paper can serve as a case study in denotational semantics.

The paper is organized as follows. In section 2 a brief introduction is given to the process algebra *ACP*. In section 3 we develop the necessary domain theory. In section 4 we give the definition of our the cpo **P** and list some of its properties. The last two sections are essentially self-contained and constitute a small overview of the domain theory relevant for our purposes. In section 5 we present our model and prove it sound.

Acknowledgement I would like to thank the Utrecht Formal Models Group for stimulating discussions.

2 The Process Algebra *ACP*

In this section we give a brief introduction to the process algebra *ACP*. For a full account of it the reader is referred to [BK84, BK85, BW91, Vaa89]. The algebra is a one-sorted algebra the elements of which are called *processes*.

Definition 2.1 *The set of terms $\mathcal{T}$ of ACP is inductively defined by*

$$s ::= a \mid \delta \mid (s;s) \mid (s+s) \mid (s \between s) \mid (s|s) \mid (s \parallel s) \mid \partial_H(s)$$

where $a \in A$ and $H \subseteq A$.

Here A is a (countable) set of constants or *primitive actions*, with typical element a. δ is a special constant that will be used to denote deadlock. The intuitive reading of the binary operators is: ; denotes sequential composition, + denotes choice, $\parallel$ denotes merge, and $\between$ and $|$ are left-merge and communication-merge, respectively. ∂_H is a unary operator for each subset $H \subseteq A$. It is the *encapsulation operator* that prevents actions $a \in H$ to be visible outside its scope. Note that we used an infix notation, as is customary. Usually we suppress brackets. It is assumed that + has lower priority than the other operators.

In this context, we are interested in the *equational theory* of the algebra, that is, in the question which terms are equal. Having the aforementioned interpretation of the operators in mind, the basic axioms of the theory are given in Table 1. We have the usual rules of transitivity, reflexivity and substitution. Note that we have assumed a function $\gamma : (A \cup \{\delta\}) \times (A \cup \{\delta\}) \to (A \cup \{\delta\})$ that is commutative, associative and

$x+y=y+x$	$A1$	$\partial_H(a)=a$ if $a \notin H$	$D1$
$x+(y+z)=(x+y)+z$	$A2$	$\partial_H(a)=\delta$ if $a \in H$	$D2$
$x+x=x$	$A3$	$\partial_H(x+y)=\partial_H(x)+\partial_H(y)$	$D3$
$(x+y);z=x;z+y;z$	$A4$	$\partial_H(x;y)=\partial_H(x);\partial_H(y)$	$D4$
$(x;y);z=x;(y;z)$	$A5$		
$x+\delta=x$	$A6$		
$\delta;x=\delta$	$A7$		
$a\|b=\gamma(a,b)$	CF		
$x \parallel y=x \between y+y \between x+x\|y$	$CM1$	$(x \between y) \between z=x \between (y \parallel z)$	$SC1$
$a \between x=a;x$	$CM2$	$(x\|y) \between z=x\|(y \between z)$	$SC2$
$(a;x) \between y=a;(x \parallel y)$	$CM3$	$x\|y=y\|x$	$SC3$
$(x+y) \between z=x \between z+y \between z$	$CM4$	$x \parallel y=y \parallel x$	$SC4$
$(a;x)\|b=(a\|b);x$	$CM5$	$x\|(y\|z)=(x\|y)\|z$	$SC5$
$a\|(b;x)=(a\|b);x$	$CM6$	$x \parallel (y \parallel z)=(x \parallel y) \parallel z$	$SC6$
$(a;x)\|(b;y)=(a\|b);(x \parallel y)$	$CM7$		
$(x+y)\|z=x\|z+y\|z$	$CM8$		
$x\|(y+z)=x\|y+x\|z$	$CM9$		

Table 1: Axioms for ACP

has δ as zero. This function encodes the basic communication between any actions $a, b \in A$: if $\gamma(a,b)=c \neq \delta$ then we say that a and b can synchronize. The synchronous execution of a and b is then regarded as the execution of the communication action c. If $\gamma(a,b)=\delta$ then a and b cannot synchronize and any attempt to synchronize them results in deadlock.

In Table 1, axioms $A1-7$ are the axioms of Basic Process Algebra with Deadlock. Axiom CF relates the communication between elementary actions a and b to our given function γ. Axioms $CM1-9$ are the axioms dealing with parallel composition. In particular, they state that the parallel composition of two processes $s_1 \parallel s_2$ performs an action by either choosing one of its arguments and performing an action of that argument, or by synchronizing. Axioms $D1-4$ deal with the Encapsulation operator. Axioms $SC1-6$ are called 'Standard Concurrency Axioms'.

Recursion is handled in this framework as follows: it just consists of adding new systems of equations like $X=s$ where s is some term possibly containing occurrences of the variable X and other variables. Formally we introduce a (countable) set $PVar$ of procedure variables, with typical element X. We extend the set of terms $\mathcal{T}$ with the extra clause $s ::= X$, that is, a procedure variable is a term. We denote this extended set of terms by $\mathcal{T}^+$.

Given a new equation $X=s$ we say that X is defined by s, or that the declaration of X is s. For technical reasons we have to insist that every term s defining some procedure variable X is guarded in X, that is, every occurence of X in s must be preceded by (at

least) one atomic action. To handle things smoothly, we require that *every* procedure variable in s is guarded. It is easy to see that the theory developed in the next sections still holds if we would employ a more liberal notion of 'guardedness'. Consult [BW91] for a fuller treatment of this notion. Our notion of guardedness amounts to introducing a sublanguage $\mathcal{T}_g \subseteq \mathcal{T}^+$ given by

$$g ::= a \mid \delta \mid (g;s) \mid (g_1 + g_2) \mid (g_1 \parallel g_2) \mid (g \between s) \mid (g_1|g_2) \mid \partial_H(g)$$

and defining $Dcl : PVar \to \mathcal{T}_g$ to be the set of admissable procedure declarations. Needless to say, an equality $s = t$ for $s, t \in \mathcal{T}^+$ only holds with respect to some fixed declaration d or, that every declaration d induces its own equality relation $=_d$.

An *interpretation* of the theory consists of a collection of elements $\mathbf{P}$ together with an interpretation function $I : \mathcal{T} \to \mathbf{P}$, assigning to each n-ary funtion symbol f, an n-ary funtion $I(f) : \mathbf{P}^n \to \mathbf{P}$. (A constant, or zero-ary function symbol is assigned an element in $\mathbf{P}$.) The interpretation extends to the whole of $\mathcal{T}$ by induction on the structure of terms. An interpretation I is called a *model* of the theory if $I(x) = I(y)$ for each axiom $x = y$ in Table 1. We can extend an interpretation I to the whole of $\mathcal{T}^+$ by assigning an element $I(X) \in \mathbf{P}$ to every procedure variable $X \in PVar$. Thus we view procedure variables as 0-ary function symbols. The interpretation I is a model for $\mathcal{T}^+$ if, moreover, $I(X) = I(d(X))$ for every $X \in PVar$. Note that such an equality in general only can be obtained if $\mathbf{P}$ admits solutions for recursively defined equations (over $\mathbf{P}$).

3 Domain Theory

In this section we give the mathematical preliminaries on which this work is based. Most of the results mentioned in this section are well-known, but appear scattered through the literature. This section introduces the various constructions and notations we use in the sequel. For a more complete treatment of the theory, consult [Plo].

Partial orders. A tuple $(D, \sqsubseteq)$ where D is a set and $\sqsubseteq \subseteq D \times D$ is a relation on D, is called a *partial order* if $\sqsubseteq$ is reflexive, transitive and anti-symmetric. We assume that each partial order has a distinguished element $\bot$ that is least with respect to the ordering relation, that is, $\bot \sqsubseteq d$ for all $d \in D$. Given a partial order $(D, \sqsubseteq)$, we call a subset $\{x_i : i < \omega\}$ a *chain* if $x_i \sqsubseteq x_{i+1}$ for all $i < \omega$. In this case we write $(x_i)_i$ for the subset. An element $d \in D$ is an *upperbound* for a chain $(x_i)_i$ if $x_i \sqsubseteq d$ for all i. d is a *least upperbound* (lub) if $d \sqsubseteq d'$ for any upperbound d'. We write $\bigsqcup_i x_i$ for the unique least upperbound of a chain $(x_i)_i$, if it exists. A partial order $(D, \sqsubseteq)$ is called *complete* (or a *cpo*) if it has least upperbounds for all chains.

The notion of chain is fundamental in the study of semantics of programming languages. Essentially, the meaning of a (recursive) program is viewed as the lub of (inductively defined) approximations to it. The approximations are all in some sense "finite". We formalize this as follows. An element $d \in D$ for some cpo D is called *finite* if $d \sqsubseteq \bigsqcup_i x_i$ implies $d \sqsubseteq x_k$ for some k. The phrase 'finite' can be understood by observing that for finite d and any chain $(x_i)_i$ such that $\bigsqcup_i x_i = d$, it must be the case that $d = x_k$ for some k. Hence any chain leading up to some finite element, stabilizes at that element. We denote the collection of all finite elements of some cpo D by $K(D)$.

Given our goal of defining a semantics, it is natural to restrict attention to cpo's that are completely determined by their finite elements. That is, for all $x \in D$ we wish there to exist a chain $(x_i)_i \subseteq K(D)$ such that $x = \bigsqcup_i x_i$. These cpo's are called *algebraic*. They are called *ω-algebraic* if moreover this collection of finite elements is a countable set. We will use the term 'domain' for an ω-algebraic cpo.

Functions. The natural notion of function between sets with some structure is of course a function that preserves the structure. In our case, the function should at least preserve the ordering. That is, if $f : D \to E$ and $x \sqsubseteq y \in D$ then $f(x) \sqsubseteq f(y) \in E$. We call these functions *monotonic*. We are quite liberal in the sense that we do not insist that functions preserve the least element. Functions f such that $f(\bot) = \bot$ are called *strict*.

The next restriction that we impose on functions is that they commute with the taking of a lub. This means that $f(\bigsqcup_i x_i) = \bigsqcup_i f(x_i)$, where the lub on the left-hand-side is taken in D, and the one on the other side in E. These functions are called *continuous*.

Since we are working with domains in which all elements arise as lub's of chains of finite elements, the continuous functions $f : D \to E$ stand in a one-to-one correspondence with monotonic functions $f' : K(D) \to E$ (*c.f.* [Kni91]). This means that every continuous function is completely determined by its action on the finite elements. This easily proven fact enables us to define a continuous function $f : D \to E$ by specifying a monotonic function $f' : K(D) \to E$, which is a much easier task.

Proposition 3.1 *Let D and E be domains. Let $f : K(D) \to E$ be monotonic and let $g : D \to E$ be monotonic and continuous. Define $\uparrow f : D \to E$ by*

$$\uparrow f(x) = \bigsqcup f(x_i)$$

where $(x_i)_i \subseteq K(D)$ is some chain such that $x = \bigsqcup_i x_i$. Then

1. *$\uparrow f$ is well-defined and continuous.*
2. *$f = (\uparrow f) \restriction K(D)$.*
3. *$g = \uparrow (g \restriction K(D))$.*

Given a function $f : D \to D$, a fixed point of f is a value $d \in D$ such that $f(d) = d$. All continuous functions have fixed points. The least fixed point of a continuous f is given by $\bigsqcup_n f^n(\bot)$. For all D, the function $fix_D : [D \to D] \to D$ that yields this least fixed point is a continuous function.

Domain constructions. First of all, observe that we can turn each countable set X into a domain $X_\bot$ by adjoining a new least element $\bot$ and stipulating that $\bot \sqsubseteq x$ and $x \sqsubseteq x$ for all $x \in X$. Cpo's of this form are called *flat*. Every set theoretic function $f : X \to Y$ can be extended to a continuous function $f_\bot : X_\bot \to Y_\bot$ by defining $f_\bot(\bot) = \bot$ and $f_\bot(x) = f(x)$ for $x \in X$.

Let D and E be domains. We define the following constructs yielding new domains:

- $D \times E$ is the cartesian product of D and E. The underlying set is

$$\{\langle x, y\rangle : x \in D, y \in E\}$$

The order is given by $\langle x_1, y_1\rangle \sqsubseteq \langle x_2, y_2\rangle$ iff $x_1 \sqsubseteq x_2$ and $y_1 \sqsubseteq y_2$. Its bottom element is $\langle \bot, \bot\rangle$.

- $D + E$ is the sum of D and E. The underlying set is

$$\{\langle 0, x\rangle : x \in D \setminus \{\bot\}\} \cup \{\langle 1, y\rangle : y \in E \setminus \{\bot\}\} \cup \{\bot\}$$

This is just the counterpart of the disjoint union of ordinary sets. For $x, y \in D+E$, the order is given by $x \sqsubseteq y$ iff $x \equiv \bot$ or $x \equiv \langle i, x'\rangle, y \equiv \langle i, y'\rangle$ and $x' \sqsubseteq y'$ $(i = 0, 1)$. Note that $D + E$ is the coproduct of D and E with respect to strict functions.

- One can generalize $+$ to arbitrary finite sums. We denote this by $D_1 + \cdots + D_n$.

If D and E are domains, then so are $D \times E$ etc. All constructions come with special functions to and from them. We define

- projections $\pi : D \times E \to D$ and $\pi' : D \times E \to E$ given by $\pi\langle x, y\rangle = x$ and $\pi'\langle x, y\rangle = y$, respectively.
- if $f : A \to D$ and $g : A \to E$ then $\langle f, g\rangle : A \to D \times E$ is given by $\langle f, g\rangle(a) = \langle f(a), g(a)\rangle$. Note that $\pi \circ \langle f, g\rangle = f$ and $\pi' \circ \langle f, g\rangle = g$.
- inclusions $in_0 : D \to D + E$ and $in_1 : E \to D + E$ given by $in_i(\bot) = \bot$ and $in_i(x) = \langle i, x\rangle$.
- sum projections $out_0 : D + E \to D$ and $out_1 : D + E \to E$ given by

$$out_0(x) = \begin{cases} \bot & \text{if } x \equiv \bot \\ \bot & \text{if } x \equiv \langle 1, y'\rangle \\ x' & \text{if } x \equiv \langle 0, x'\rangle \end{cases}$$

and likewise for out_1.

- sum discriminators $is_0 : D + E \to \mathbf{T}$ and $is_1 : D + E \to \mathbf{T}$ given by

$$is_0(x) = \begin{cases} \bot & \text{if } x \equiv \bot \\ \mathbf{t} & \text{if } x \equiv \langle 0, x'\rangle \\ \mathbf{f} & \text{if } x \equiv \langle 1, x'\rangle \end{cases}$$

and likewise for is_1. Here we have used the domain of truthvalues $\mathbf{T} = \{\bot, \mathbf{t}, \mathbf{f}\}$ with the obvious partial ordering.

- all functions can be extended to finite sums.

Finally we review the construction of the powerdomain $\mathcal{P}^*(D)$. It is more difficult than the previous ones, and a more detailed exposition can be found in [Kni91]. Given a domain D we want to define $\mathcal{P}^*(D)$. We start with the collection $\mathcal{F}(D)$ of all finite sets of finite elements of D. These will act as the finite elements of $\mathcal{P}^*(D)$. We order $\mathcal{F}(D)$ by putting

$$X \sqsubseteq_{EM} Y \text{ iff } (\forall x \in X \exists y \in Y. x \sqsubseteq y) \wedge (\forall y \in Y \exists x \in X. x \sqsubseteq y)$$

This order is called the Egli-Milner order. $(\mathcal{F}(D), \sqsubseteq_{EM})$ is a pre-ordered set, so we can form its completion to get a domain (see [Kni91]). This domain is by definition the powerdomain of D. A suitable representation uses the following closure operation.

$$Cl(X) = \{\bigsqcup x_i \subseteq K(D) : \exists x \in X.x \sqsubseteq \bigsqcup x_i \ \& \ \forall i \exists x' \in X.x_i \sqsubseteq x'\}$$

It is easy to check that Cl indeed is a set theoretical closure operation. Furthermore we have $Cl(X) = Con(X)$ for all $X \in \mathcal{F}(D)$ where $Con(X) = \{y : \exists x_1, x_2 \in X.x_1 \sqsubseteq y \sqsubseteq x_2\}$ is the convex closure operator. Defining

$$Up(X_i)_i = \{\bigsqcup x_i : x_i \in X_i\}$$

for a Egli-Milner ordered chain $(X_i)_i$ in $\mathcal{F}(D)$, the powerdomain then consists of all sets $Cl(Up(X_i)_i)$ for chains $(X_i)_i \subseteq \mathcal{F}(D)$. Unfortunately, the ordering of these sets is in general no longer Egli-Milner, but becomes the Plotkin order. For details, consult [Plo76, Smy78, Kni91]. We will always be working with finite elements, and these are Egli-Milner ordered.

- For $f : D \to E$ we define $\mathcal{P}^*f : \mathcal{P}^*(D) \to \mathcal{P}^*(E)$ by

  $$\mathcal{P}^*f(X) = Cl\{f(x) : x \in X\}$$

- For $f : D_1 \times D_2 \to E$ we define $f^\dagger : \mathcal{P}^*(D_1) \times D_2 \to \mathcal{P}^*(E)$ by

  $$f^\dagger(X, y) = Cl\{f(x, y) : x \in X\}$$

- For $f : D_1 \times D_2 \to E$ we define $f^\ddagger : \mathcal{P}^*(D_1) \times \mathcal{P}^*(D_2) \to \mathcal{P}^*(E)$ by

  $$f^\dagger(X_1, X_2) = Cl\{f(x_1, x_2) : x_1 \in X_1, x_2 \in X_2\}$$

- We have a continuous function $\uplus : \mathcal{P}^*(D) \times \mathcal{P}^*(D) \to \mathcal{P}^*(D)$ given by

  $$\uplus(X, Y) = Cl(X \cup Y)$$

- We have a continuous function $\{\!|\cdot|\!\} : D \to \mathcal{P}^*(D)$ given by

  $$\{\!|x|\!\} = \{x\}$$

- We have a continuous function $\biguplus : \mathcal{P}^*(\mathcal{P}^*(D)) \to \mathcal{P}^*(D)$ given by

  $$\biguplus(X) = Cl\left(\bigcup X\right)$$

The constructions $(\cdot) \times (\cdot)$, $\mathcal{P}^*(\cdot)$ etc., are all continuous. That is, for a chain of functions $(f_i)_i$ we have $g \times \bigsqcup_i f_i = \bigsqcup_i (g \times f_i)$ etc. In particular this means that one can solve recursive domain equations involving these constructors (*c.f.* [Plo, SP82]). Also, the derived operations $(\cdot)^\dagger$ and $(\cdot)^\ddagger$ are continuous.

A function $f : \mathcal{P}^*(D) \to \mathcal{P}^*(E)$ is called *linear* if $f(X \uplus Y) = f(X) \uplus f(Y)$ for all $X, Y \in \mathcal{P}^*(D)$. It is known that $\mathcal{P}^* f$, $(f)^\dagger$ and $(f)^\ddagger$ are linear.

We need the following continuous function **if** $(\cdot)$ **then** $(\cdot)$ **else** $(\cdot) : \mathbf{T} \times D \times D \to D$ given by

$$\textbf{if } t \textbf{ then } d_1 \textbf{ else } t_2 = \begin{cases} \bot & \text{if } t = \bot \\ d_1 & \text{if } t = \mathbf{t} \\ d_2 & \text{if } t = \mathbf{f} \end{cases}$$

4 The Semantic Domain

In this section we give the definition of the domain **P** that underlies our order-theoretic denotational semantics for *ACP*. **P** is defined as the least solution of the following reflexive equation

$$\mathbf{P} \cong \mathcal{P}^*(\{\delta\}_\bot + A_\bot + A_\bot \times \mathbf{P})$$

where $\delta \notin A$ is used to denote deadlock. For definiteness, let ϕ be the isomorphism between the left- and right-hand sides.

It is instructive to see how the solution **P** is obtained as we will need the construction in the sequel. For a full treatment of the theory of solving reflexive domain equations, see [Plo, SP82]. The following theory is taken from those papers. Briefly, we solve the equation in $\mathbf{Cpo}^E$, the category that has as objects cpo's and as arrows *embedding-projection pairs*

$$(k : D \to E, l : E \to D)$$

satisfying

$$l \circ k = 1_D \quad k \circ l \sqsubseteq 1_E.$$

Now **P** is the colimit of the sequence

$$P_0 = \{\bot\}$$

$$P_{n+1} = \mathcal{P}^*(\{\delta\}_\bot + A_\bot + A_\bot \times P_n)$$

with as embedding-projection pairs the pairs $(i_n : P_n \to P_{n+1}, j_n : P_{n+1} \to P_n)$ given by

$$i_0 = j_0 = \lambda x.\bot$$

$$i_{n+1} = \mathcal{P}^*(1_{\{\delta\}} + 1_A + 1_A \times i_n)$$

$$j_{n+1} = \mathcal{P}^*(1_{\{\delta\}} + 1_A + 1_A \times j_n)$$

Intuitively, i_n is the inclusion of P_n into P_{n+1} and j_n maps a set at nesting depth $n+1$ onto $\perp$. We write

$$i_{nm} = i_{m-1} \circ \cdots \circ i_{n+1} \circ i_n$$

$$j_{nm} = j_n \circ j_{n+1} \circ \cdots \circ j_{m-1}$$

The colimit comes equipped with functions $\alpha_n : P_n \to \mathbf{P}$ and $\beta_n : \mathbf{P} \to P_n$. These α_n and β_n have the properties that $\alpha_n \circ \beta_n \sqsubseteq \alpha_{n+1} \circ \beta_{n+1}$, $\bigsqcup_n \alpha_n \circ \beta_n = 1_{\mathbf{P}}$, $\alpha_n = \alpha_{n+1} \circ i_n$ and $\beta_n = j_n \circ \beta_{n+1}$.

Concretely, $\mathbf{P}$ consists of ω-indexed sequences

$$p \equiv \langle p_0, p_1, \ldots, p_n, \ldots \rangle$$

where $p_n \in P_n$ such that $p_n = j_n(p_{n+1})$. Then

$$\beta_n(\langle p_0, \ldots, p_n, \ldots \rangle) = p_n$$

and

$$\alpha_n(p) = \langle j_{0n}(p), \ldots, j_{(n-1)n}(p), p, i_{n(n+1)}(p), \ldots \rangle$$

for $p \in P_n$. As $\{\delta\}_\perp$ and $A_\perp$ are ω-algebraic, every P_n is ω-algebraic. In turn, $\mathbf{P}$ itself is ω-algebraic and its collection of finite elements is given by $\alpha_n(p)$ for $p \in K(P_n)$.

We now give a characterization of the continuous functions $f : \mathbf{P} \to \mathbf{P}$ in terms of functions $f_n : P_n \to P_n$. This characterization will enable us to derive properties of functions $f : \mathbf{P} \to \mathbf{P}$ by showing that these properties hold of certain (induced) functions $f^{(n)} : P_n \to P_n$ and invoking a limit argument. The latter task will be substantially easier since we are allowed to reason by inductive arguments. To our knowledge, this approach is new.

First of all, to each $f : \mathbf{P} \to \mathbf{P}$ we associate a function $f^{(n)} : P_n \to P_n$ by $f^{(n)} = \beta_n \circ f \circ \alpha_n$ for each n. Observe that we have

$$\begin{aligned} j_n \circ f^{(n+1)} \circ i_n &= j_n \circ \beta_{n+1} \circ f \circ \alpha_{n+1} \circ i_n \\ &= \beta_n \circ f \circ \alpha_n \\ &= f^{(n)} \end{aligned}$$

Let us call a family of functions $\{f_n : P_n \to P_n : n < \omega\}$ ***compatible*** if $f_n = j_n \circ f_{n+1} \circ i_n$. We write $[f_n]_n$ for such a family. Note that a family of functions is compatible if $f_n \circ j_n = j_n \circ f_{n+1}$.

Each $f_n : P_n \to P_n$ gives rise to a function $\bar{f}_n : \mathbf{P} \to \mathbf{P}$ given by $\bar{f}_n = \alpha_n \circ f_n \circ \beta_n$. For a compatible family $[f_n]_n$ we have

$$\begin{aligned} \bar{f}_n &= \alpha_n \circ j_n \circ f_{n+1} \circ i_n \circ \beta_n \\ &\sqsubseteq \alpha_{n+1} \circ f_{n+1} \circ \beta_{n+1} \\ &= \bar{f}_{n+1} \end{aligned}$$

Hence we may define $[f_n]_n^\updownarrow = \bigsqcup_n \bar{f}_n$. This is a continuous function.

Proposition 4.1 *For all* $f : \mathbf{P} \to \mathbf{P}$ *and all compatible families* $[f_n]_n$ *we have*

1. $[f^{(n)}]_n$ *is compatible.*
2. $[f_n]_n^{\uparrow}$ *is continuous.*
3. $[f^{(n)}]_n^{\uparrow} = f$.
4. $\forall n. \left([f_n]_n^{\uparrow}\right)^{(n)} = f_n$.

Proof We have already proven 1. and 2. above. For 3. we have

$$\begin{aligned}
[f^{(n)}]_n^{\uparrow} &= \bigsqcup \alpha_n \circ \beta_n \circ f \circ \alpha_n \circ \beta_n \\
&= \bigsqcup (\alpha_n \circ \beta_n) \circ f \circ \bigsqcup (\alpha_n \circ \beta_n) \\
&= 1_{\mathbf{P}} \circ f \circ 1_{\mathbf{P}} \\
&= f
\end{aligned}$$

For 4. we have

$$\begin{aligned}
\left([f_n]_n^{\uparrow}\right)^{(n)} &= \beta_n \circ [f_n]_n^{\uparrow} \circ \alpha_n \\
&= \beta_n \circ \bigsqcup (\alpha_m \circ f_m \circ \beta_m) \circ \alpha_n \\
&= \bigsqcup (\beta_n \circ \alpha_m \circ f_m \circ \beta_m \circ \alpha_n) \\
&= \bigsqcup (j_{nm} \circ f_m \circ i_{nm}) \\
&= f_n
\end{aligned}$$

□

5 The Denotational Semantics

In this section we define our denotational semantics $[\![-]\!]$ and prove that this semantics provides a model for *ACP*. To this end we define semantic functions corresponding to the syntactic constructors of the language.

Sequential composition We want to define a function $\odot : \mathbf{P} \times \mathbf{P} \to \mathbf{P}$. First define the higher order operator Ψ

$$\Psi : [\mathbf{P} \times \mathbf{P} \to \mathbf{P}] \to [(\{\delta\}_\perp + A_\perp + A_\perp \times \mathbf{P}) \times \mathbf{P} \to (\{\delta\}_\perp + A_\perp + A_\perp \times \mathbf{P})]$$

by

$$\begin{aligned}
\Psi(f)(x,p) = \ & \textbf{if } is_0(x) \\
& \textbf{then } x \\
& \textbf{else if } is_1(x) \\
& \qquad \textbf{then } in_2(\langle out_1(x), p\rangle) \\
& \qquad \textbf{else } in_2(\langle \pi(out_2(x)), f(\pi'(out_2(x)), p)\rangle)
\end{aligned}$$

and $\tilde{\Psi} = \lambda f. \phi^{-1} \circ (\Psi_I(f))^{\dagger} \circ (\phi \times 1)$. Put $\odot = fix(\tilde{\Psi})$.

In the remainder of this section, we suppress the functions in_i and out_i for the sake of simplicity, trusting that the reader can provide them himself where needed. For instance, we will write $\{\!|a|\!\}$ instead of $\{\!|in_1(a)|\!\}$. This convention increases readability of the expressions considerably.

Lemma 5.1 *The compatible family* $[\odot^{(n)}]_n$ *is given by the following functions,*

$$\odot_n = (\tilde{\odot}_n)^\dagger$$

where we use the auxiliary construct

$$\tilde{\odot}_{n+1} : [\{\delta\}_\perp + A_\perp + A_\perp \times P_n] \times P_{n+1} \to [\{\delta\}_\perp + A_\perp + A_\perp \times P_n]$$

inductively given by

$$\tilde{\odot}_0 = \lambda x, y.\perp$$

$$\begin{aligned}\tilde{\odot}_{n+1} = \lambda x, y.\ & \textbf{if } is_0(x)\\ & \textbf{then } x\\ & \textbf{else if } is_1(x)\\ & \quad \textbf{then } \langle x, j_n(y)\rangle\\ & \quad \textbf{else } \langle \pi(x), \pi'(x) \odot_n j_n(y)\rangle\end{aligned}$$

Proof First we observe that we can define α_n and β_n with the help of functions $\tilde{\alpha}_n$ inductively given by

$$\begin{aligned}&\textbf{if } is_0(x) \vee is_1(x)\\ &\textbf{then } x\\ &\textbf{else } \langle \pi(x), \alpha_n(\pi'(x))\rangle\end{aligned}$$

putting $\alpha_n = \phi^{-1} \circ \mathcal{P}^*(\tilde{\alpha}_n) \circ \phi$. Likewise for β_n. Hence both α_n and β_n are linear.

We proceed by induction on n to show that $\odot^{(n)} = \odot_n$. The case $n = 0$ is trivial. For $n \geq 0$, we have $p_1 \odot^{(n+1)} p_2 = \beta_{n+1}(\alpha_{n+1}(p_1) \odot \alpha_{n+1}(p_2))$. This last function is the pointwise extension of

$$\begin{aligned}&\textbf{if } is_0(x)\\ &\textbf{then } x\\ &\textbf{else } \textbf{if } is_1(x)\\ &\quad \textbf{then } \langle x, \beta_n(\alpha_{n+1}(p_2))\rangle\\ &\quad \textbf{else } \langle \pi(x), \beta_n(\alpha_n(\pi'(x)) \odot \alpha_{n+1}(p_2))\rangle\end{aligned}$$

We have that $\beta_n \circ \alpha_{n+1} = j_n$ and that

$$\beta_n(p \odot \alpha_m(p')) = \beta_n(p \odot \alpha_n(j_{mn}(p')))$$

for all $n, m \geq n$. Hence $\beta_n(\alpha_n(\pi'(x)) \odot \alpha_{n+1}(p_2)) = \beta_n(\alpha_n(\pi'(x)) \odot \alpha_n(j_n(p_2)))$. Hence the desired conclusion. □

Corollary 5.2 $\odot^{(n)} \circ (j_n \times j_n) = j_n \circ \odot^{(n+1)}$.

Lemma 5.3 *For all* $p_1, p_2, p_3 \in \mathbf{P}$,

1. $(p_1 \odot p_2) \odot p_3 = p_1 \odot (p_2 \odot p_3)$ *(Axiom A5);*
2. $\{\!|in_0(\delta)|\!\} \odot p_1 = \{\!|in_0(\delta)|\!\}$ *(Axiom A6).*

Proof First we show that for all n, $(p'_1 \odot^{(n)} p'_2) \odot^{(n)} p'_3 = p'_1 \odot^{(n)} (p'_2 \odot^{(n)} p'_3)$ for all $p'_1, p'_2, p'_3 \in P_n$. The case $n = 0$ is trivial. For the induction step we argue as follows. The right hand side of the equation, considered as a function of p'_1, p'_2, p'_3, is the pointwise extension of

$$\begin{array}{l}\textbf{if } is_0(x)\\ \textbf{then } x\\ \textbf{else } \textbf{if } is_1(x)\\ \qquad \textbf{then } \langle x, j_n(p'_2 \odot^{(n+1)} p'_3)\rangle\\ \qquad \textbf{else } \langle \pi(x), \pi'(x) \odot^{(n)} j_n(p'_2 \odot^{(n+1)} p'_3)\rangle\end{array}$$

By properties of the compatible family $[\odot^{(n)}]_n$, we have that

$$\pi'(x) \odot^{(n)} j_n(p'_2 \odot^{(n+1)} p'_3) = \pi'(x) \odot^{(n)} (j_n(p'_2) \odot^{(n)} j_n(p'_3))$$

By induction hypothesis, the last expression equals

$$(\pi'(x) \odot^{(n)} j_n(p'_2)) \odot^{(n)} j_n(p'_3)$$

Now it is easy to see that the left-hand side is the pointwise extension of the aforementioned function.

The claim now follows because

$$\begin{aligned}(p_1 \odot p_2) \odot p_3 &= (\bigsqcup_n \alpha_n(\beta_n(p_1) \odot^{(n)} \beta_n(p_2))) \odot p_3\\ &= \bigsqcup_m \alpha_m(\beta_m(\bigsqcup_n \alpha_n(\beta_n(p_1) \odot^{(n)} \beta_n(p_2)))) \odot^{(m)} \beta_m(p_3))\\ &= \bigsqcup_m \alpha_m(\bigsqcup_{n>m} j_{nm}(\beta_n(p_1) \odot^{(n)} \beta_n(p_2))) \odot^{(m)} \beta_m(p_3))\\ &= \bigsqcup_m \alpha_m((\beta_m(p_1) \odot^{(m)} \beta_m(p_2)) \odot^{(m)} \beta_m(p_3))\\ &= \bigsqcup_m \alpha_m(\beta_m(p_1) \odot^{(m)} (\beta_m(p_2) \odot^{(m)} \beta_m(p_3)))\\ &= p_1 \odot (p_2 \odot p_3)\end{aligned}$$

The second equality is immediate. □

In order to define the other operators it is convenient to have another function Δ available. This function $\Delta : \mathbf{P} \to \mathbf{P}$ removes δ's from the first level of its argument, unless that argument equals $\{\!|\delta|\!\}$. Note that this function is of a 'global' nature, that is, Δ is not linear. It is this definition that enables us to formulate our model. We can define $\Delta_n : P_n \to P_n$ uniformly on the finite elements of P_n by

$$\Delta_n(X) = \begin{cases} X & \text{if } X \equiv \{\langle 0, \delta\rangle\} \\ X \setminus \{\langle 0, \delta\rangle\} & \text{otherwise}\end{cases}$$

It is easy to see that Δ_n is monotonic on the finite elements of P_n and hence extends to a continuous function on P_n. It is also easy to show that $\Delta_n \circ j_n = j_n \circ \Delta_{n+1}$, hence $[\Delta_n]_n$ is a compatible family. Let $\Delta : \mathbf{P} \to \mathbf{P}$ be the induced function on $\mathbf{P}$. The following lemmas list some elementary properties of Δ.

Lemma 5.4 *For all* $p, p' \in \mathbf{P}$,

1. $\Delta(\Delta(p)) = \Delta(p)$;
2. $\Delta(p \uplus p') = \Delta(\Delta(p) \uplus \Delta(p')) = \Delta(p \uplus \Delta(p'))$;
3. $\Delta(p \odot p') = \Delta(p) \odot p'$.

Choice We define the semantic choice operator $\oplus : \mathbf{P} \times \mathbf{P} \to \mathbf{P}$ as $\oplus = \Delta \circ \uplus$.

Remark. Note that with this definition of the choice operator, it is in general *not* the case that $p \oplus p = p$ or that $p \oplus \{\!|\delta|\!\} = p$. We can define another choice operator $\hat{\oplus}$ such that these equalities hold for all $p \in \mathbf{P}$ with help of $\hat{\oplus}'$ given by

$$
\begin{array}{l}
\textbf{if } is_0(x) \\
\textbf{then } \{\!|y|\!\} \\
\textbf{else } \textbf{if } is_0(y) \\
\qquad \textbf{then } \{\!|x|\!\} \\
\qquad \textbf{else } \{\!|x|\!\} \uplus \{\!|y|\!\}
\end{array}
$$

and setting $\hat{\oplus} = \uplus \circ (\hat{\oplus}')^{\ddagger}$. Proofs of the properties of other operators would become more difficult, however. Also, the present formulation has a strong underlying intuition as a *global choice* operator and gives rise to the identification of the (relevant) substructure $\mathbf{P}'$ (see below). Furthermore, we need Δ in the definition of various operators below.

Lemma 5.5 *For all* $p_1, p_2, p_3 \in \mathbf{P}$,

1. $p_1 \oplus p_2 = p_2 \oplus p_1$ *(Axiom A1)*;
2. $p_1 \oplus (p_1 \oplus p_3) = (p_1 \oplus p_2) \oplus p_3$ *(Axiom A2)*;
3. $(p_1 \oplus p_2) \odot p_3 = (p_1 \odot p_3) \oplus (p_2 \odot p_3)$ *(Axiom A4)*.

Proof The first equality is trivial. For the other two equalities, we have

$$
\begin{array}{rcl}
p_1 \oplus (p_2 \oplus p_3) & = & \Delta(p_1 \uplus \Delta(p_2 \uplus p_3)) \\
& = & \Delta(p_1 \uplus p_2 \uplus p_3) \\
& = & \Delta(\Delta(p_1 \uplus p_2) \uplus p_3) \\
& = & (p_1 \oplus p_2) \oplus p_3 \\
(p_1 \oplus p_2) \odot p_3 & = & \Delta(p_1 \uplus p_2) \odot p_3 \\
& = & \Delta((p_1 \uplus p_2) \odot p_3) \\
& \overset{(*)}{=} & \Delta(p_1 \odot p_3 \uplus p_2 \odot p_3) \\
& = & (p_1 \odot p_3) \oplus (p_2 \odot p_3)
\end{array}
$$

where equality $(*)$ holds since $\odot$ is defined pointwise in its first argument and hence is linear in that argument. □

Parallel composition We define the parallel composition operator $\otimes : \mathbf{P} \times \mathbf{P} \to \mathbf{P}$ as follows.

First recall the higher order operator Ψ used in the definition of sequential composition. This operator will again be used in the definition of the left-merge operator. For communications, we lift the function γ as given in the definition of the algebra to

$$\gamma : (\{\delta\}_\perp + A_\perp) \times (\{\delta\}_\perp + A_\perp) \to (\{\delta\}_\perp + A_\perp)$$

Hence γ is strict, commutative, and associative and has $\langle 0, \delta \rangle$ as zero. We now define

$$\begin{aligned} \Psi_C \;:\; & [\mathbf{P} \times \mathbf{P} \to \mathbf{P}] \to \\ & [(\{\delta\}_\perp + A_\perp + A_\perp \times \mathbf{P}) \times (\{\delta\}_\perp + A_\perp + A_\perp \times \mathbf{P}) \to (\{\delta\}_\perp + A_\perp + A_\perp \times \mathbf{P})] \end{aligned}$$

by

$$\begin{aligned} \Psi_C(f)(x,y) = \;& \textbf{if } (is_0(x) \vee is_1(x)) \wedge (is_0(y) \vee is_1(y)) \\ & \textbf{then } \gamma(x,y) \\ & \textbf{else if } is_1(x) \wedge is_2(y) \\ & \quad \textbf{then } \textbf{if } is_0(\gamma(x,\pi(y))) \\ & \qquad \textbf{then } \{\!|\delta|\!\} \\ & \qquad \textbf{else } \langle \gamma(x,\pi(y)), \pi'(y) \rangle \\ & \quad \textbf{else } \textbf{if } is_2(x) \wedge is_1(y) \\ & \qquad \textbf{then } \textbf{if } is_0(\gamma(\pi(x),y)) \\ & \qquad\quad \textbf{then } \{\!|\delta|\!\} \\ & \qquad\quad \textbf{else } \langle \gamma(\pi(x),y), \pi'(x) \rangle \\ & \qquad \textbf{else if } is_0(\gamma(\pi(x),\pi(y))) \\ & \qquad\quad \textbf{then } \{\!|\delta|\!\} \\ & \qquad\quad \textbf{else } \langle \gamma(\pi(x),\pi(y)), f(\pi'(x),\pi'(y)) \rangle \end{aligned}$$

and we define $\tilde{\Psi}_C = \lambda f.\Delta \circ \phi^{-1} \circ (\Psi_C(f))^\ddagger \circ (\phi \times \phi)$.

We now define

$$\Phi(f)(p_1,p_2) = \tilde{\Psi}(f)(p_1,p_2) \oplus \tilde{\Psi}(f)(p_2,p_1) \oplus \tilde{\Psi}_C(f)(p_1,p_2)$$

and put $\otimes = fix(\Phi)$. Likewise, we define $\otimes_L = \tilde{\Psi}(\otimes)$ and $\mathbin{⦶} = \tilde{\Psi}_C(\otimes)$. The next lemma follows immediately from the definitions.

Lemma 5.6 *For all $a, b, c \in A$, $p, p' \in \mathbf{P}$,*

1. $\{\!|a|\!\} \mathbin{⦶} \{\!|b|\!\} = \{\!|b|\!\} \mathbin{⦶} \{\!|a|\!\} = \{\!|\gamma(a,b)|\!\}$ *(Axiom CF);*
2. $(\{\!|a|\!\} \mathbin{⦶} \{\!|b|\!\}) \mathbin{⦶} \{\!|c|\!\} = \{\!|a|\!\} \mathbin{⦶} (\{\!|b|\!\} \mathbin{⦶} \{\!|c|\!\}) = \{\!|\gamma(\gamma(a,b),c)|\!\}$;
3. $\{\!|a|\!\} \otimes_L p = \{\!|\langle a,p \rangle|\!\} = \{\!|a|\!\} \odot p$ *(Axiom CM2);*
4. $(\{\!|a|\!\} \odot p) \otimes_L p' = \{\!|a|\!\} \odot (p \otimes p')$ *(Axiom CM3);*

5. $(\{\!|a|\!\} \odot p) \mathbin{\textcircled{\scriptsize D}} \{\!|b|\!\} = \{\!|\gamma(a,b)|\!\} \odot p$ *(Axiom CM5);*

6. $\{\!|a|\!\} \mathbin{\textcircled{\scriptsize D}} (\{\!|b|\!\} \odot p) = \{\!|\gamma(a,b)|\!\} \odot p$ *(Axiom CM6);*

7. $(\{\!|a|\!\} \odot p) \mathbin{\textcircled{\scriptsize D}} (\{\!|b|\!\} \odot p') = \{\!|\gamma(a,b)|\!\} \odot (p \otimes p')$ *(Axiom CM7).*

Lemma 5.7 *For all* $p_1, p_2, p_3 \in \mathbf{P}$,

1. $p_1 \otimes p_2 = (p_1 \otimes_L p_2) \oplus (p_2 \otimes_L p_1) \oplus (p_1 \mathbin{\textcircled{\scriptsize D}} p_2)$ *(Axiom CM1);*

2. $(p_1 \oplus p_2) \otimes_L p_3 = (p_1 \otimes_L p_3) \oplus (p_2 \otimes_L p_3)$ *(Axiom CM4);*

3. $(p_1 \oplus p_2) \mathbin{\textcircled{\scriptsize D}} p_3 = (p_1 \mathbin{\textcircled{\scriptsize D}} p_3) \oplus (p_2 \mathbin{\textcircled{\scriptsize D}} p_3)$ *(Axiom CM8);*

4. $p_1 \mathbin{\textcircled{\scriptsize D}} (p_2 \oplus p_3) = (p_1 \mathbin{\textcircled{\scriptsize D}} p_2) \oplus (p_1 \mathbin{\textcircled{\scriptsize D}} p_3)$ *(Axiom CM9).*

Proof For the second equality, we have $\Delta(p \otimes_L p') = \Delta(p) \otimes_L p'$ just like for $\odot$. The claim now follows by the same argument. For the third equality, Δ is the left-most function in the definition of $\mathbin{\textcircled{\scriptsize D}}$. Hence

$$\begin{aligned}(p_1 \oplus p_2) \mathbin{\textcircled{\scriptsize D}} p_3 &= \Delta(p_1 \uplus p_2) \mathbin{\textcircled{\scriptsize D}} p_3 \\ &= (p_1 \uplus p_2) \mathbin{\textcircled{\scriptsize D}} p_3 \\ &\overset{(*)}{=} \Delta((p_1 \mathbin{\textcircled{\scriptsize D}} p_3) \uplus (p_2 \mathbin{\textcircled{\scriptsize D}} p_3)) \\ &= (p_1 \mathbin{\textcircled{\scriptsize D}} p_3) \oplus (p_2 \mathbin{\textcircled{\scriptsize D}} p_3)\end{aligned}$$

where equality $(*)$ holds since $\mathbin{\textcircled{\scriptsize D}}$ is by definition linear. The fourth equality is proved similarly. □

Next we are going to show that the Standard Concurrency Axioms hold. First, we have that $\otimes^{(n)}$ is given by:

$$\otimes^{(n)}(p,p') = \otimes_L^{(n)}(p,p') \oplus^{(n)} \otimes_L^{(n)}(p',p) \oplus^{(n)} \mathbin{\textcircled{\scriptsize D}}^{(n)}(p,p')$$

where $\mathbin{\textcircled{\scriptsize D}}^{(n)}$ is inductively given by $\tilde{\Phi}^{(0)}(x,y) = \bot$ and $\tilde{\Phi}^{(n+1)} = \Psi_C(\mathbin{\textcircled{\scriptsize D}}^{(n)})$ putting $\mathbin{\textcircled{\scriptsize D}}^{(n+1)} = \Delta \circ (\tilde{\Phi}^{(n+1)})^\sharp$. Furthermore, $\otimes_L^{(n)}$ is given completely analogous to $\odot^{(n)}$.

$$\tilde{\otimes}_L^{(0)} = \lambda x, y.\bot$$

$$\begin{aligned}\tilde{\otimes}_L^{(n+1)} = \lambda x, y.\ &\textbf{if } is_0(x) \textbf{then } x \\ &\textbf{else if } is_1(x) \textbf{then } \langle x, j_n(y)\rangle \\ &\quad\textbf{else } \langle \pi(x), \pi'(x) \otimes^{(n)} j_n(y)\rangle\end{aligned}$$

Lemma 5.8 *For all* $p, p' \in \mathbf{P}$,

1. $p \mathbin{\textcircled{\scriptsize D}} p' = p' \mathbin{\textcircled{\scriptsize D}} p$ *(Axiom SC3);*

2. $p \otimes p' = p' \otimes p$ *(Axiom SC4).*

Proof We prove that, for all $p, p' \in P_n$, $p \oplus^{(n)} p' = p' \oplus^{(n)} p$ and $p \otimes^{(n)} p' = p' \otimes^{(n)} p$ by simultaneous induction on n. The first equality follows easily from the fact that $\tilde{\oplus}^{(n)}$ is defined symmetrically in x and y, and the induction hypothesis. The claim for $\otimes^{(n)}$ is then trivial. □

Lemma 5.9 *For all* $p_1, p_2, p_3 \in \mathbf{P}$,

1. $(p_1 \otimes_L p_2) \otimes_L p_3) = p_1 \otimes_L (p_2 \otimes p_3)$ *(Axiom SC1);*
2. $(p_1 \oplus p_2) \otimes_L p_3 = p_1 \oplus (p_2 \otimes_L p_3)$ *(Axiom SC2);*
3. $p_1 \oplus (p_2 \oplus p_3) = (p_1 \oplus p_2) \oplus p_3$ *(Axiom SC5);*
4. $p_1 \otimes (p_2 \otimes p_3) = (p_1 \otimes p_2) \otimes p_3$ *(Axiom SC6).*

Proof We prove the equalities in their $(\cdot)^{(n)}$ form, by simultaneous induction on n. The base case $n = 0$ is in all cases trivial. For the first equality, we have

$$\begin{aligned}
\tilde{\otimes}_L^{(n+1)}(x, \otimes^{(n+1)}(p_2, p_3)) \;=\; & \textbf{if } is_0(x) \\
& \textbf{then } x \\
& \textbf{else } \textbf{if } is_1(x) \\
& \qquad \textbf{then } \langle x, j_n(\otimes^{(n+1)}(p_2, p_3))\rangle \\
& \qquad \textbf{else } \langle \pi(x), \otimes^{(n)}(\pi'(x), j_n(\otimes^{(n+1)}(p_2, p_3)))\rangle
\end{aligned}$$

$$\begin{aligned}
\tilde{\otimes}_L^{(n+1)}(\tilde{\otimes}_L^{(n+1)}(x, p_2), p_3) \;=\; & \textbf{if } is_0(x) \\
& \textbf{then } x \\
& \textbf{else } \textbf{if } is_1(x) \\
& \qquad \textbf{then } \langle x, \otimes^{(n)}(j_n(p_2), j_n(p_3))\rangle \\
& \qquad \textbf{else } \langle \pi(x), \otimes^{(n)}(\otimes^{(n)}(\pi'(x), j_n(p_2)), j_n(p_3))\rangle
\end{aligned}$$

By properties of the compatible family $[\otimes^{(n)}]_n$ and the induction hypothesis on $\otimes^{(n)}$ these two expressions are the same.

The second equality is proved likewise.

The associativity of $\oplus^{(n+1)}$ follows readily from the associativity of γ and the induction hypothesis on $\otimes^{(n)}$.

For the last equality, we write out left- and right-hand-sides of the equality.

$$\begin{aligned}
& p_1 \otimes^{(n+1)} (p_2 \otimes^{(n+1)} p_3) = \\
& \quad = [p_1 \otimes_L^{(n+1)} (p_2 \otimes^{(n+1)} p_3)]^{\{1\}} \oplus^{(n+1)} [(p_2 \otimes_L^{(n+1)} p_3) \otimes_L^{(n+1)} p_1]^{\{2\}} \oplus^{(n+1)} \\
& \quad\; [(p_3 \otimes_L^{(n+1)} p_2) \otimes_L^{(n+1)} p_1]^{\{3\}} \oplus^{(n+1)} [(p_2 \oplus^{(n+1)} p_3) \otimes_L^{(n+1)} p_1]^{\{4\}} \oplus^{(n+1)} \\
& \quad\; [p_1 \oplus^{(n+1)} (p_2 \otimes_L^{(n+1)} p_3)]^{\{5\}} \oplus^{(n+1)} [p_1 \oplus^{(n+1)} (p_3 \otimes_L^{(n+1)} p_2)]^{\{6\}} \oplus^{(n+1)} \\
& \quad\; [p_1 \oplus^{(n+1)} (p_2 \oplus^{(n+1)} p_3)]^{\{7\}}.
\end{aligned}$$

$$(p_1 \otimes^{(n+1)} p_2) \otimes^{(n+1)} p_3 =$$

$$
\begin{aligned}
= \; & [(p_1 \otimes_L^{(n+1)} p_2) \otimes_L^{(n+1)} p_3]^{\{1'\}} \oplus^{(n+1)} [(p_2 \otimes_L^{(n+1)} p_1) \otimes_L^{(n+1)} p_3]^{\{2'\}} \oplus^{(n+1)} \\
& [(p_1 \oplus^{(n+1)} p_2) \otimes_L^{(n+1)} p_3]^{\{5'\}} \oplus^{(n+1)} [p_3 \otimes_L^{(n+1)} (p_1 \otimes^{(n+1)} p_2)]^{\{3'\}} \oplus^{(n+1)} \\
& [(p_1 \otimes_L^{(n+1)} p_2) \oplus^{(n+1)} p_3]^{\{6'\}} \oplus^{(n+1)} [(p_2 \otimes_L^{(n+1)} p_1) \oplus^{(n+1)} p_3]^{\{4'\}} \oplus^{(n+1)} \\
& [(p_1 \oplus^{(n+1)} p_2) \oplus^{(n+1)} p_3]^{\{7'\}}.
\end{aligned}
$$

In the above we have, by the previous cases in the lemma, that $\{i\} = \{i'\}$ for $1 \leq i \leq 7$. □

Encapsulation Each subset $H \subseteq A$ gives rise to the following continuous function $\widehat{H} : A_\perp \to \mathbf{T}$:

$$
\widehat{H}(x) = \begin{cases} \perp & \text{if } x \equiv \perp \\ \mathbf{t} & \text{if } x \not\equiv \perp \;\&\; x \in H \\ \mathbf{f} & \text{otherwise} \end{cases}
$$

Fixing such a set $H \subseteq A$, we define $\nabla_H : \mathbf{P} \to \mathbf{P}$ with the help of

$$
\Psi_H : (\mathbf{P} \to \mathbf{P}) \to (\{\delta\}_\perp + A_\perp + A_\perp \times \mathbf{P}) \to (\{\delta\}_\perp + A_\perp + A_\perp \times \mathbf{P})
$$

by

$$
\begin{aligned}
\Psi_H(f)(x) = \; & \textbf{if } is_1(x) \\
& \textbf{then } \textbf{if } \widehat{H}(x) \\
& \qquad \textbf{then } \delta \\
& \qquad \textbf{else } x \\
& \textbf{else if } is_2(x) \\
& \qquad \textbf{then } \textbf{if } \widehat{H}(\pi(x)) \\
& \qquad\qquad \textbf{then } \delta \\
& \qquad\qquad \textbf{else } \langle \pi(x), f(\pi'(x)) \rangle \\
& \qquad \textbf{else } x
\end{aligned}
$$

Now define $\tilde{\Psi}_H = \lambda f.\Delta \circ \phi^{-1} \circ \mathcal{P}^*(\Psi_H(f)) \circ \phi$ and $\nabla_H = fix(\tilde{\Psi}_H)$. We give some elementary properties of ∇_H.

Lemma 5.10 *Let $H \subseteq A$ and let $a \in A$. Then*

1. $\nabla_H(\{\!|a|\!\}) = \{\!|\delta|\!\}$ *if* $a \in H$ *(Axiom D1);*
2. $\nabla_H(\{\!|a|\!\}) = \{\!|a|\!\}$ *if* $a \notin H$ *(Axiom D2).*

Lemma 5.11 *For all $p, p' \in \mathbf{P}$,*

1. $\nabla_H(p) = \nabla_H(\Delta(p)) = \Delta(\nabla_H(p))$;
2. $\nabla_H(p \uplus p') = \Delta(\nabla_H(p) \uplus \nabla_H(p'))$;
3. $\nabla_H(p \oplus p') = \nabla_H(p) \oplus \nabla_H(p')$ *(Axiom D3).*

Proof We have

$$\begin{aligned}\nabla_H(p \oplus p') &= \nabla_H(\Delta(p \uplus p')) \\ &= \Delta(\nabla_H(p) \uplus \nabla_H(p')) \\ &= \nabla_H(p) \oplus \nabla_H(p')\end{aligned}$$

The other equalities follow immediately from the definitions. □

Lemma 5.12 *For all* $p_1, p_2 \in \mathbf{P}$, $\nabla_H(p_1 \odot p_2) = \nabla_H(p_1) \odot \nabla_H(p_2)$ *(Axiom D4).*

Proof Again define the compatible family $\{\nabla_H^{(n)} : P_n \to P_n : n < \omega\}$ and use induction on n to prove that for each n,

$$\nabla_H^{(n)}(p \odot^{(n)} p') = \nabla_H^{(n)}(p) \odot^{(n)} \nabla_H^{(n)}(p')$$

for all $p, p' \in P_n$. □

Having defined semantical counterparts to all syntactic operators, we are ready to define the denotational semantics. First of all, let $\Gamma : PVar \to \mathbf{P}$ be the set of *environments* or meanings of procedure variables.

Definition 5.13 *We define* $D : \mathcal{L} \to \Gamma \to \mathbf{P}$ *by induction on the structure of* s *as follows:*

- $D(\gamma)(a) = \{\!|a|\!\}$;
- $D(\gamma)(\delta) = \{\!|\delta|\!\}$;
- $D(\gamma)(s_1 \cdot s_2) = D(\gamma)(s_1) \odot D(\gamma)(s_2)$;
- $D(\gamma)(s_1 + s_2) = D(\gamma)(s_1) \oplus D(\gamma)(s_2)$;
- $D(\gamma)(s_1 \between s_2) = D(\gamma)(s_1) \otimes_L D(\gamma)(s_2)$;
- $D(\gamma)(s_1 | s_2) = D(\gamma)(s_1) \between D(\gamma)(s_2)$;
- $D(\gamma)(s_1 \parallel s_2) = D(\gamma)(s_1) \otimes D(\gamma)(s_2)$;
- $D(\gamma)(\partial_H(s)) = \nabla_H(D(\gamma)(s))$;
- $D(\gamma)(X) = \gamma(X)$.

The order on $\mathbf{P}$ extends to an order on Γ. The higher-order operator $\Upsilon : \Gamma \to \Gamma$ defined by

$$\Upsilon(\gamma)(X) = D(\gamma)(d(X))$$

is a continuous operator and hence has a fixed point γ_d. Now define $[\![-]\!] : \mathcal{L} \to \mathbf{P}$ as $D(\gamma_d)$.

In view of the preceding lemmas, the only equalities that we still need to prove are Axioms $A3$ and $A6$. We can define the following subset $\mathbf{P}' \subset \mathbf{P}$. First, define the operator $\Delta^* : \mathbf{P} \to \mathbf{P}$ as

$$\begin{aligned}\Theta(f)(x) = \ & \textbf{if } is_0(x) \vee is_1(x) \\ & \textbf{then } x \\ & \textbf{else } \langle \pi(x), f(\pi'(x)) \rangle\end{aligned}$$

and set $\tilde{\Theta} = \lambda f.\Delta \circ \phi^{-1} \circ \mathcal{P}^*(\Theta(f)) \circ \phi$. Set $\Delta^* = fix(\tilde{\Theta})$. Intuitively, Δ^* applies Δ recursively to a process. Define $\mathbf{P}' = \Delta^*(\mathbf{P})$, the direct image of $\mathbf{P}$ under Δ^*. As $\Delta^*(\Delta^*(p)) = \Delta^*(p)$, $\mathbf{P}'$ consists of all fixed points of Δ^*. For the next proposition we need the following definition. Given a domain D, a subset $D' \subseteq D$ is called *inclusive* if whenever $(x_i)_i \subseteq D'$ then $\bigsqcup x_i \in D'$.

Proposition 5.14 $\mathbf{P}'$ *is an inclusive subset of* $\mathbf{P}$.

Proof Given a chain $(p_i)_i \subseteq \mathbf{P}'$, we have that

$$\Delta^*(\bigsqcup p_i) = \bigsqcup \Delta^*(p_i) = \bigsqcup p_i$$

Hence $\mathbf{P}'$ is closed under lub's. □

Since $\bot, [\![a]\!], [\![\delta]\!] \in \mathbf{P}'$ and all operators preserve the property of being in $\mathbf{P}'$, it follows that $[\![-]\!] : \mathcal{T}^+ \to \mathbf{P}'$.

Lemma 5.15 *For all* $p \in \mathbf{P}'$,

1. $p \oplus p = p$ *(Axiom A3);*
2. $p \oplus \{\!|\delta|\!\} = p$ *(Axiom A6).*

Hence we have obtained the following, main result of the paper:

Theorem 5.16 $[\![-]\!]$ *is a model for ACP.*

6 Completeness

In the preceding section we saw that $x = y$ in the equational theory implies $[\![x]\!] = [\![y]\!]$. The question arises whether the semantics is *complete* in the sense that the reverse implication also holds. In this section we give an affirmative answer to that question, thus showing that the model we gave for *ACP* is complete. In the sequel we will denote an equation $x = y$ of the theory by $\vdash x = y$.

First of all, we need some results from the theory of *ACP*. For a fuller treatment and proofs of the facts mentioned here, consult [BW91].

Definition 6.1 *The collection $\mathcal{N}$ of* normal forms *of terms is given by the following grammar.*

$$n ::= \delta \mid \sum_{i=1}^{n} a_i \mid \sum_{i=1}^{n} a_i + \sum_{k=1}^{m} a_k \cdot n_i$$

where $\sum_{i=1}^{1} x \equiv x$ and $a_i \in A$.

For any $x \in \mathcal{T}$ there exists a unique $x' \in \mathcal{N}$ such that $\vdash x = x'$. Hence we have a function $\mathcal{NF} : \mathcal{T} \to \mathcal{N}$ assigning to each term its unique normal form. We have the following proposotion.

Proposition 6.2 *For all $x, y \in \mathcal{T}$, $\vdash x = y$ iff $\vdash \mathcal{NF}(x) = \mathcal{NF}(y)$.*

We can assign to each total $p \in P_n' = \Delta^*(P_n)$ the following element $\mathcal{NF}(p) \in \mathcal{N}$. For $n = 1$, let $p \in P_n'$ be total, that is, p has no occurrence of $\perp$. Then either $p \equiv \{\langle 0, \delta\rangle\}$, or $p \equiv \{\langle 1, a_1\rangle, \ldots, \langle 1, a_n\rangle\}$. In the first case, $\mathcal{NF}(p) = \delta$ and in the second case, $\mathcal{NF}(p) = \sum_{i=1}^{n} a_i$. For the induction step in the definition, let $p \in P_{n+1}'$ be total. Then $p \equiv \{\langle 0, \delta\rangle\}$, or $p \equiv \{\langle 1, a_1\rangle, \ldots, \langle 1, a_n\rangle, \langle 2, \langle a_1, p_1\rangle\rangle, \ldots, \langle 2, \langle a_m, p_m\rangle\rangle\}$ where $p_k \in P_n'$. In the first case, $\mathcal{NF}(p) = \delta$ and in the second case $\mathcal{NF}(p) = \sum_{i=1}^{n} a_i + \sum_{k=1}^{m} a_m \cdot \mathcal{NF}(p_k)$. Note that we have to "choose" an order in which to list the elements of p in the definition of $\mathcal{NF}$. In view of Axioms $A1$ and $A2$, however, this order is irrelevant. Hence for a process p, if $\mathcal{NF}'$ is defined analogously to $\mathcal{NF}$ but with another listing of the elements of p, we have $\vdash \mathcal{NF}(p) = \mathcal{NF}'(p)$.

The following proposition shows that the functions $\mathcal{NF}$ and $[\![-]\!]$ are inverse to each other.

Proposition 6.3 1. *For all total $p \in \bigcup_{n<\omega} P_n'$, $p = [\![\mathcal{NF}(p)]\!]$.*

2. *For all $x \in \mathcal{N}$, $\vdash x = \mathcal{NF}([\![x]\!])$.*

Corollary 6.4 *For all $x_1, x_2 \in \mathcal{N}$, $\vdash x_1 = x_2$ iff $[\![x_1]\!] = [\![x_2]\!]$.*

Theorem 6.5 *For all $x, y \in \mathcal{T}$, $\vdash x = y$ iff $[\![x]\!] = [\![y]\!]$.*

Proof We have

$$\begin{aligned} \vdash x = y \quad &\text{iff} \quad \vdash \mathcal{NF}(x) = \mathcal{NF}(y) \\ &\text{iff} \quad [\![\mathcal{NF}(x)]\!] = [\![\mathcal{NF}(y)]\!] \\ &\text{iff} \quad [\![x]\!] = [\![y]\!] \end{aligned}$$

where the latter equivalence holds since, by soundness, we have that $[\![x]\!] = [\![\mathcal{NF}(x)]\!]$. □

In order to extend the previous Theorem to the whole of $\mathcal{T}^+$ we need some means to relate infinite terms. This is dealt with using projections π_n and the Approximation

Induction Principle which says that $x = y$ iff $\pi_n(x) = \pi_n(y)$ for all n. Here π_n is inductively defined as

$$\begin{aligned}
\pi_n(a) &= a \\
\pi_n(\delta) &= \delta \\
\pi_1(a \cdot x) &= a \\
\pi_{n+1}(a \cdot x) &= a \cdot \pi_n(x) \\
\pi_n(x + y) &= \pi_n(x) + \pi_n(y)
\end{aligned}$$

Note that we now have defined projections on the set of normal forms $\mathcal{N}$. These projections extend to the whole of $\mathcal{T}^+$ by the following procedure. Let $t_1 \in \mathcal{T}^+$. Given t_i define t_{i+1} to be the term in which every procedure variable X is replaced by its body $d(X)$. Obtain the normal form $\mathcal{NF}(t_n)$ by considering each procedure variable X occurring in t_n as a (new) constant action. Then put $\pi_n(t_1) = \pi_n(\mathcal{NF}(t_n))$.

We can define projections $\pi_n : \mathbf{P} \to P_n$ as follows. Define $\tilde{\pi}_n$ by

$$\begin{aligned}
\tilde{\pi}_1(x) &= \textbf{if } is_0(x) \vee is_1(x) \textbf{ then } x \\
&\quad\ \textbf{else } \pi(x) \\
\tilde{\pi}_{n+1}(x) &= \textbf{if } is_0(x) \vee is_1(x) \textbf{ then } x \\
&\quad\ \textbf{else } \langle \pi(x), \pi_n(\pi'(x)) \rangle
\end{aligned}$$

and set $\pi_n = \mathcal{P}^*(\tilde{\pi}_n)$.

We have the following lemma.

Lemma 6.6 *For all $x \in \mathcal{T}^+$, $[\![\pi_n(x)]\!] = \pi_n[\![x]\!]$.*

Proof The lemma holds obviously for $x \in \mathcal{N}$. For $x \in \mathcal{T}$, we have $[\![x]\!] = [\![\mathcal{NF}(x)]\!]$. Hence $[\![\pi_n(x)]\!] = [\![\pi_n(\mathcal{NF}(x))]\!] = \pi_n[\![\mathcal{NF}(x)]\!]$. For $x \in \mathcal{T}^+$ we first observe $\vdash \pi_n(x) = \pi_n(x')$ where x' is the n^{th} expansion of x in which every procedure variable is replaced by an arbitrary term $s \in \mathcal{T}$. This follows easily from the fact that we have guarded recursion. Likewise, $\pi_n[\![x]\!] = \pi_n[\![x']\!]$. Hence $[\![\pi_n(x)]\!] = [\![\pi_n(x')]\!] = \pi_n[\![x']\!] = \pi_n[\![x]\!]$. □

Obviously, π_n and β_n are closely related. We have the following proposition.

Proposition 6.7 *For all n, $p.p' \in \mathbf{P}$,*

1. *$\beta_n(p) = \beta_n(p')$ implies $\pi_n(p) = \pi_n(p')$;*
2. *$\pi_{n+1}(p) = \pi_{n+1}(p')$ implies $\beta_n(p) = \beta_n(p')$.*

Corollary 6.8 *For all $p, p' \in \mathbf{P}$, $p = p'$ iff for all n, $\pi_n(p) = \pi_n(p')$.*

Theorem 6.9 *For all $x, y \in \mathcal{T}^+$, $\vdash x = y$ iff $[\![x]\!] = [\![y]\!]$.*

Proof

$$\begin{aligned}
\vdash x = y \quad &\text{iff} \quad \forall n. \vdash \pi_n(x) = \pi_n(y) \\
&\text{iff} \quad \forall n. [\![\pi_n(x)]\!] = [\![\pi_n(y)]\!] \\
&\text{iff} \quad \forall n. \pi_n[\![x]\!] = \pi_n[\![y]\!] \\
&\text{iff} \quad [\![x]\!] = [\![y]\!].
\end{aligned}$$

□

7 Discussion

We have defined an order-theoretic interpretation for *ACP* and we have proved that it is a model. Some remarks seem in place. First of all, traditionally *ACP* is interpreted over so-called process-graphs modulo strong bisimulation. A canonical model for the algebra of finite terms (*i.e.* without procedure variables) is the set of finitely branching trees of finite depth. These trees precisely correspond to finite and *total* (*i.e.* not containing $\perp$) elements in $\mathbf{P}'$, onto which $[\![-]\!]$ maps these terms. It follows that our semantics is also complete, at least for the finite terms. That is, if $t_1 \neq t_2$ then $[\![t_1]\!] \neq [\![t_2]\!]$. Next, a recursively specified process X can be given meaning in the *projective limit model*. This is essentially similar to how the denotation of X is obtained in $\mathbf{P}$.

As remarked in the introduction, $\mathbf{P}$ contains a lot of 'junk', *i.e.* elements $p \notin \mathbf{P}'$. Moreover, $\mathbf{P}'$ itself contains junk: consider *e.g.* the infinite process $\{\langle 1, a\rangle : a \in A\} \cup \{\perp\}$. Since our recursion is guarded, this process cannot be obtained as the denotation of any term in $\mathcal{T}$.

An important feature of *ACP* is the inclusion in the algebra of a so-called *silent move* τ, which is used to model a step or a sequence of steps local to a process. Axioms for τ include $\tau \cdot x + x = \tau \cdot x$. It is readily seen that this axiom prevents us from modeling the semantic choice operator by a continuous function. Hence we cannot extend our model to cover this extension of the algebra. A possible way out is to model τ as an ordinary atomic action and then divide out the equivalence induced by the new axioms. It would be interesting to pursue this possibility.

References

[BK84] J.A. Bergstra and J.W. Klop. Process algebra for synchronous communication. *Information and Control*, 60:109–137, 1984.

[BK85] J.A. Bergstra and J.W. Klop. Algebra for communicating processes with abstraction. *Theor. Comp. Sc.*, 37(1):77–121, 1985.

[BW91] J. Baeten and P. Weyland. *Process Algebra.* Cambridge Tracts in Theoretical Computer Science. Cambridge Univ. Press, 1991.

[Hoa85] C.A.R. Hoare. *Communicating Sequential Processes.* Prentice-Hall International, 1985.

[Kni91] P.M.W. Knijnenburg. Algebraic domains, chain completion and the powerdomain construction. Technical report, Utrecht Univ., 1991.

[LS86] J. Lambek and P.J. Scott. *Introduction to Higher Order Categorical Logic.* Cambridge Studies in Advanced Mathematics. Cambridge Univ. Press, 1986.

[Mil80] R. Milner. *A Calculus of Communicating Systems.* Lecture Notes in Computer Science, Vol. 92. Springer-Verlag, Berlin, 1980.

[Plo] G.D. Plotkin. Pisa lecture notes. Unpublished.

[Plo76] G.D. Plotkin. A powerdomain construction. *SIAM J. on Computing*, 5:452–487, 1976.

[Smy78] M.B. Smyth. Power domains. *J. of Comp. Syst. Sc.*, 16:23–36, 1978.

[SP82] M.B. Smyth and G.D. Plotkin. The category-theoretic solution of recursive domain equations. *SIAM J. Comput.*, 11(4):761–783, 1982.

[Vaa89] F.W. Vaandrager. *Algebraic techniques for concurrency and their applications*. PhD thesis, Universiteit van Amsterdam, 1989.