Reihe Informatik

30/95

# Denotational linear time semantics and sequential composition

C. Baier, M.E. Majster-Cederbaum

# Denotational Linear Time Semantics and Sequential Composition

Christel Baier, Mila E. Majster-Cederbaum

Fakultät für Mathematik und Informatik
Universität Mannheim, 68131 Mannheim
{baier,mcb}@pi1.informatik.uni-mannheim.de

December 1995

**Abstract**

This paper focuses on the issue of modelling sequential composition in denotational linear time semantics for (nondeterministic) languages which admit infinite computations. This operator deserves special attention as it causes problems to meet the requirements of a standard denotational semantics based on metric or cpo. We present a general framework for the treatment of sequential composition. It turns out that a program can be described by its maximal computations in the metric approach whereas the partial order approach is suitable to describe a program by all its partial computations.

## Contents

# 1 Introduction

Usually the semantic domain of linear time semantics is a collection of subsets of a semantic domain $A$ where the elements of $A$ can be considered as computations of programs. The meaning of a program $P$ is then a subset $H$ of $A$ where the elements of $H$ correspond to the possible computations of $P$. Typical examples are trace [5, 11] or pomset semantics [4, 7, 15].

Two kinds of computations can be distinguished: maximal and partial computations. Maximal computations can either be infinite or finite. The latter include successfull terminating computations as well as deadlocked computations. Partial computations are finite execution fragments of maximal computations. A partial computation leads either to a final state or to an intermediate state, i.e. a state in which the computation goes on. In other words, partial computations are either terminating computations or computations which can be extended to maximal computations.

One attempt of this paper is to present conditions which characterize 'good' sequential operators on the underlying domains. By a 'good' sequential operator we mean an operator which reflects the ideas of sequential composition as specified by a given operational semantics. If $P$ and $Q$ are programs then their sequential composition $P;Q$ is a program which first behaves like $P$ and if $P$ has successfull terminated then it behaves like $Q$. In this paper we do not deal with deacklocked processes, i.e. we assume that termination is always successfull. Hence the set of maximal computations of $P;Q$ consists of the infinite computations of $P$ and all computations which start with a terminating computation of $P$ followed by a maximal computation of $Q$. The set of partial computations of $P;Q$ consists of the partial computations of $P$ (possibly except for those terminating computations which pronounce their termination) and all computations which first behave like a terminating computation of $P$ and then perform a partial computation of $Q$. If $A$ is a semantic domain whose elements can be interpreted as maximal resp. partial computations then a 'good' sequential operator $;_A$ on $A$ would satisfy:

(i) If $x \in A$ is an infinite computation resp. a computation leading to an intermediate state then $x ;_A y = x$.

(ii) If $x$ is a terminating computation then $x ;_A y$ stands for a computation which first performs $x$ and then $y$.

Such an operator on $A$ induces the operator $(H, I) \mapsto \{ x ;_A y : x \in H, y \in I \}$ on the powerset of $A$. Hence our aim is find conditions which ensure that a semantic operator on $A$ for modelling sequential composition satisfies the conditions (i) and (ii). (i) and (ii) imply that $A$ has to distinguish between elements representing terminating computations from those which stand for infinite computations resp. computations leading to an intermediate state. Dealing with maximal computations it seems to be natural to assume that

the elements of $A$ representing infinite computations differ from the elements of $A$ which stand for terminating (i.e. finite) computations. In contrast to this, dealing with partial computations it might be the case that in $A$ terminating computations cannot be distinguished from partial computations leading to an intermediate state. For instance, if $A$ is the set of finite strings over some action alphabet *Act* where a string $\alpha_1 \ldots \alpha_n$ is considered as a computation which performs successively the actions $\alpha_1, \ldots, \alpha_n$ then the terminating computation which performs $\alpha_1, \ldots, \alpha_n$ and then stops cannot be distinguished from the computation which performs $\alpha_1, \ldots, \alpha_n$ but does not come to a halt. We generalize the idea of [11] and model termination by a new action $\surd$. We show how a metric setting can be used to associate with a process $P$ the set of its maximal computations, and how the cpo setting can be used to give a partial computation meaning.

The paper is organized as follows: Section 2 presents the syntax of the language *Prog* which is under consideration for the whole paper. Section 3 presents conditions which allow the definition of a metric denotational linear time semantics which assigns to each program the set of its maximal computations. In section 4 we argue that the partial order approach fails to describe the maximal computations. Hence we switch to partial computations and give a denotational linear time semantics which maps each program to the set of its partial computations. In section 5 we show how the partial computations of a program which are given by a partial order semantics as in section 4 can be derived from its maximal computations which are given by a metric semantics as in section 3. The assumption that the elements of a semantic domain $A$ can be interpreted as partial computations implies that $A$ is (or can be) equipped with a partial order: $x \sqsubseteq_A y$ iff $x$ is an execution fragment of $y$. For this reason we do not discuss the question whether the metric approach works for partial computations in absense of a suitable partial order. Section 6 contains some concluding remarks. Throughout the whole paper we deal with traces and pomsets as applications of our framework.

# 2 The language *Prog*

Throughout we consider a language *Prog* which includes nondeterministic choice, sequential composition and recursion. We assume a fixed set *Act* of atomic actions $\alpha, \beta, \ldots$. Each action $\alpha$ represents a program that performs $\alpha$ and then stops. In addition to the binary operator symbols $+$ and $;$ which stand for nondeterministic choice resp. sequential composition we assume a set $\Omega$ of operator symbols for modelling operators like parallelism with or without communication, relabelling, hiding, and so on. Each operator symbol $\omega$ is associated with an arity $|\omega| \geq 1$. Recursion is modelled by guarded declarations, i.e. we assume a fixed set *Idf* of identifiers and a fixed mapping $\sigma$ that assigns a guarded statement $\sigma(\xi)$ to each identifier $\xi$. Formally, the set $Prog^{\sigma}(\Omega)$ (or *Prog* for short) of programs is given by the production system

$$P ::= \alpha \mid \xi \mid P_1 + P_2 \mid P_1; P_2 \mid \omega(P_1, \ldots, P_k)$$

where $\alpha \in Act$, $\xi \in Idf$ and where $\omega \in \Omega$ is a $k$-ary operator symbol. Guarded statements are given by the production system:

$$G ::= \alpha \mid G_1 + G_2 \mid G; P \mid \omega(G_1, \ldots, G_k)$$

$\sigma$ is a fixed mapping from $Idf$ into the set of guarded statements. Each occurrence of an identifier $\xi$ in a program $P \in Prog$ is a recursive call of the procedure $\sigma(\xi)$. The guardedness of the statements $\sigma(\xi)$ is essential for the definition of a denotational semantics in the metric approach. This assumption can be omitted if one only wants to give a denotational semantics in the partial order approach.

**Example 2.1** In the following we will consider the language $Prog^\sigma(\{\|\})$ as a standard examle which is given by the production system

$$P \ ::= \ \alpha \mid \xi \mid P_1 + P_2 \mid P_1 \,;\, P_2 \mid P_1 \| P_2$$

where $\|$ denotes parallel composition without communication. As before we assume $\sigma$ to be a fixed guarded declaration. An operational semantics for $Prog^\sigma(\{\|\})$ can be given by as in [5]: Let $E$ be a new symbol. Then

$$\rightarrow \subseteq Prog^\sigma(\{\|\}) \times Act \times (Prog^\sigma(\{\|\}) \cup \{E\})$$

is the smallest relation which satisfies the following conditions (where we write $P \stackrel{\alpha}{\rightarrow} P'$ instead of $(P, \alpha, P') \in \rightarrow$):

$\alpha \stackrel{\alpha}{\rightarrow} E$

$$\frac{P_1 \stackrel{\alpha}{\rightarrow} Q}{P_1 + P_2 \stackrel{\alpha}{\rightarrow} Q} \qquad \frac{P_2 \stackrel{\alpha}{\rightarrow} Q}{P_1 + P_2 \stackrel{\alpha}{\rightarrow} Q} \quad \text{where } Q \in Prog^\sigma(\{\|\}) \cup \{E\}$$

$$\frac{P_1 \stackrel{\alpha}{\rightarrow} P_1'}{P_1 \,;\, P_2 \stackrel{\alpha}{\rightarrow} P_1' \,;\, P_2} \qquad \frac{P_1 \stackrel{\alpha}{\rightarrow} E}{P_1 \,;\, P_2 \stackrel{\alpha}{\rightarrow} P_2} \quad \text{where } P_1' \in Prog^\sigma(\{\|\})$$

$$\frac{P_1 \stackrel{\alpha}{\rightarrow} P_1'}{P_1 \| P_2 \stackrel{\alpha}{\rightarrow} P_1' \| P_2} \qquad \frac{P_2 \stackrel{\alpha}{\rightarrow} P_2'}{P_1 \,;\, P_2 \stackrel{\alpha}{\rightarrow} P_1 \| P_2'} \quad \text{where } P_1', P_2' \in Prog^\sigma(\{\|\})$$

$$\frac{P_1 \stackrel{\alpha}{\rightarrow} E}{P_1 \| P_2 \stackrel{\alpha}{\rightarrow} P_2} \qquad \frac{P_2 \stackrel{\alpha}{\rightarrow} E}{P_1 \| P_2 \stackrel{\alpha}{\rightarrow} P_1}$$

$$\frac{\sigma(\xi) \stackrel{\alpha}{\rightarrow} Q}{\xi \stackrel{\alpha}{\rightarrow} Q} \quad \text{where } Q \in Prog^\sigma(\{\|\}) \cup \{E\}$$

We consider the labelled transition system $(Prog^\sigma(\{\|\}) \cup \{E\}, \rightarrow)$. Computations of a program $P$ are reflected in the paths starting at $P$. A maximal finite computation of $P$ corresponds to a path from $P$ to $E$. □

A standard technique to define a denotational linear time semantics for a language of type $Prog^\sigma(\Omega)$ consists of two steps:

(i) provide a semantic domain $A$ to model the finite deterministic behaviour, i.e. the sublanguage $Prog_{\text{fin}}$ given by

$$P \ ::= \ \alpha \mid P_1; P_2 \mid \omega(P_1, \ldots, P_k)$$

together with suitable semantic operatots $\alpha_A \in A$, $;_A : A \times A \to A$ and $\omega_A : A^k \to \mathcal{P}_{\text{fin}}(A)$ where $\mathcal{P}_{\text{fin}}(A)$ denotes the collection of nonempty and finite subsets of $A$. The meaning function is then

$$Me^A : Prog_{\text{fin}} \to \mathcal{P}_{\text{fin}}(A)$$

with

$$\begin{aligned}
Me^A(\alpha) &= \{\alpha_A\} \\
Me^A(P_1 + P_2) &= Me^A(P_1) \cup Me^A(P_2) \\
Me^A(P_1; P_2) &= \left\{ x ;_A y : x \in Me^A(P_1),\ y \in Me^A(P_2) \right\} \\
Me^A(\omega(P_1, \ldots, P_k)) &= \bigcup \left\{ \omega_A(x_1, \ldots, x_k) : x_i \in Me^A(P_i),\ i = 1, \ldots, k \right\}
\end{aligned}$$

(ii) extend this semantics $Me^A$ to infinite computations where some powerdomain construction $\mathcal{P}_*(A)$ of $A$ is used as semantic domain. Since the computations of $P + Q$ are those of $P$ and $Q$ the nondeterministic choice operator $+$ has to be modelled by the union on $\mathcal{P}_*(A)$.

Throughout we consider the semantic domains of traces and pomsets as standard examples where the underlying language is $Prog^\sigma(\{\|\})$. The semantic operators for modelling nondeterministic choice, sequential composition and parallelism are those of [11] and [5] in the case of traces and those of [4] and [7] in the case of pomsets.

**Example 2.2** Let $Act^*$ be the set of finite sequences over $Act$ (which we call traces). $Act^+$ denotes the subset of nonempty finite traces. The atomic action $\alpha \in Act$ is associated with the single-element trace $\alpha$. Sequential composition is interpreted by concatenation:

$$x; y \stackrel{\text{def}}{=} xy$$

Parallelism is modelled by interleaving, i.e. $x \parallel y$ is the set of traces which arise by merging:

$$x \parallel y \stackrel{\text{def}}{=} x \lceil y \cup y \lceil x$$

where $\lceil$ means leftmerge, i.e. $\emptyset \lceil z \stackrel{\text{def}}{=} \{z\}$, $\alpha x \lceil z \stackrel{\text{def}}{=} \{ \alpha w : w \in x \lceil z \}$. E.g. the interpretation of the program $P = (\alpha; \gamma) \parallel \beta$ as a subset of $Act^*$ is

$$\{ \alpha\gamma\beta,\ \alpha\beta\gamma, \beta\alpha\gamma \}.$$

□

**Example 2.3** A pomset is a partially ordered set $(E, \leq)$ which is endowed with a labelling function $l : E \to Act$ that maps the elements of $E$ (called events) to some action. The interpretation of pomsets as computations is as follows: The execution of an event $e \in E$ means the execution of the associated action $l(e)$. If $e < e'$ (i.e. $e \leq e'$ and $e \neq e'$) then $e$ must be executed before $e'$. If $e$, $e'$ are independent events (i.e. neither $e \leq e'$ nor

$e' \leq e$) then $e$ and $e'$ may be executed in parallel. In addition we require that each event is reachable, i.e. for each $e \in E$ the set of predecessors of $e$ is finite.

$Pom^*$ denotes the set of isomorphism classes of finite pomsets, $Pom^+$ the subset of nonempty and finite pomsets. Here by a finite pomset we mean a pomset where the underlying partially ordered set is finite. Isomorphism of pomsets means that they only differ in the names of the events, i.e. we abstract from the names of the events. In what follows, we identify pomsets and their isomorphism classes.

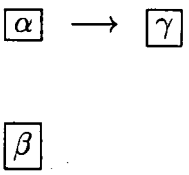
The associated pomset of an atomic action $\alpha$ is the pomset $p_\alpha$ which consists of a single event labelled by $\alpha$. Given two finite pomsets $p_1$, $p_2$ we get $p_1; p_2$ by appending $p_2$ at the end of $p_1$, i.e. each event of $p_2$ is preceeded by all events of $p_1$. Formally, sequential composition ; on $Pom^*$ is given by:

$$(E_1, \leq_1, l_1) \; ; \; (E_2, \leq_2, l_2) \stackrel{\text{def}}{=} (E_1 \cup E_2, \leq, l_1 \cup l_2)$$

where $E_1 \cap E_2 = \emptyset$ and $\leq \; = \; \leq_1 \cup \leq_2 \cup E_1 \times E_2$. The parallel execution $p_1 \parallel p_2$ of two pomsets is modelled by taken the disjoint union, i.e. for all events $e$ of $p_1$ and $e'$ of $p_2$ the events $e$ and $e'$ are independent in $p_1 \parallel p_2$.

$$(E_1, \leq_1, l_1) \parallel (E_2, \leq_2, l_2) \stackrel{\text{def}}{=} (E_1 \cup E_2, \leq_1 \cup \leq_2, l_1 \cup l_2)$$

where $E_1$, $E_2$ are disjoint. E.g. the meaning of the program $S \; = \; (\alpha; \gamma) \parallel \beta$ is then the set consisting of the pomset:




□

# 3 Denotational linear time semantics in the metric approach

In this section we show how a denotational linear time semantics defined in the metric approach can be given such that the meaning of a program can be viewed the set of its maximal computations. The starting point is a metric space $M$ suitable to model finite behaviour. The completion $\overline{M}$ of $M$ is used to describe finite and infinite behaviour. We present conditions which the semantic operators on $M$ have to fulfill and show how under these conditions suitable semantic operators on $\mathcal{P}_{co}(\overline{M})$, the collection of all nonempty and compact subsets of $\overline{M}$, can be derived thus yielding a denotational semantics on $\mathcal{P}_{co}(\overline{M})$.

## 3.1 Metric spaces suitable to model finite behaviour

We assume that the metric on $M$ is a measure for the number of atomic steps in which two computations agree. We require that for each two computations $x$, $y \in M$:

$$d(x, y) \; = \; \frac{1}{2^n}$$

if and only if $x$ and $y$ coincide in the first $n$ steps and differ in the $(n+1)$-th step. The interpretation of a 'step' depends on the underlying semantic domain. A step might be the execution of a single atomic action $\alpha \in Act$, the communication of two (or more) atomic actions or the parallel execution of some atomic acions and/or communications between atomic actions. We formalize this assumption by the concept of a ranking. The notion of a ranking as we use it here is closely related to the notions of a rank ordering or a projection space as they were introduced in [8] resp. [9].

**Definition 3.1** *A* ranking *on a metric space $M$ is a sequence of functions*

$$M \to M, \quad x \mapsto x[n]$$

*where $n$ ranges over the natural numbers $\geq 1$ such that:*

- $(x[n])[m] \;=\; (x[m])[n] \;=\; x[n]$ *for all* $m \geq n \geq 1$
- *for each $x \in M$ there is some $n \geq 1$ with $x = x[n]$*
- *the metric $d$ on $M$ is given by the formula*

$$d(x,y) \;=\; \inf \left\{ \frac{1}{2^n} \;:\; x[n] = y[n] \right\}$$

*where* $\inf \emptyset = 1$. *We put*

$$\rho(x) \;=\; \min \{\; n \;:x[n] = x \;\}.$$

*If $H$ is a subset of $M$, $\tilde{x} \;=\; (x_1, \ldots, x_k) \;\in\; M^k$ then we put:*

$$H[n] \;=\; \{\; x[n] \;:\; x \in H \;\}, \quad \tilde{x}[n] \;=\; (x_1[n], \ldots, x_k[n])$$

We interpret $\rho(x)$ as the number of steps which the computation $x$ performs. The second conditions ensures that each element of $M$ can be considered as a finite (i.e. terminating) computation. In addition, it implies that the elements of $M$ stand for 'nonempty' computations, i.e. computations that perform at least one step. $x[n]$ is called the $n$-cut of $x$. $x[n]$ represents the behaviour of $x$ until the $n$-th step. This is due to the fact that $\rho(x[n]) \leq n$ and $d(x, x[n]) \leq 1/2^n$. We have $\rho(x) \leq n$ if and only if $x = x[n]$. If $\rho(x) > n$ then we may think of $x[n]$ as a process which behaves like $x$ in the first $n$ steps and then terminates.

**Example 3.2** The metric on $Act^+$

$$d(x,y) \;=\; \inf \left\{ \frac{1}{2^n} \;:\; n\text{-th prefix of } x \;=\; n\text{-th prefix of } y \right\}$$

is induced by the ranking $x \mapsto x[n]$ where $x[n]$ denotes the $n$-th prefix of $x$. Then $\rho(x)$ is the usual length of the string $x$. The metric of [4] on $Pom^+$ is given by

$$d(p,q) \;=\; \left\{ \frac{1}{2^n} \;:\; p[n] \;=\; q[n] \right\}$$

is induced by the ranking $p \mapsto p[n]$. Here $p[n]$ is the pomset which arises from $p$ by removing all events $e$ with $depth_p(e) > n$ where

$$depth_p(e) \;=\; \max \{\; n \;: \exists e_1, \ldots, e_n \in E \;\; e_1 < \ldots < e_n = e \;\}.$$

Then $\rho(p)$ is the usual depth of $p$:

$$depth(p) \;=\; \max \left\{\; depth_p(e) \;:\; e \text{ is an event of } p \;\right\}$$

□

In the following definition we formalize the properties which an operator for modelling sequential composition has to fulfill. If $\rho(x) \geq n$ (i.e. the execution of $x$ needs at least $n$ steps) then the first $n$ steps of $x; y$ equal those of $x$. If $\rho(x) = m < n$ (i.e. the execution of $x$ stops after performing $m$ steps) then the first $n$ steps of $x; y$ consist of the execution of $x$ followed by the first $n - m$ steps of $y$.

**Definition 3.3** *Let $M$ be a metric space which is equipped with a ranking. An operator*

$$M \times M \to M, \;\; (x, y) \;\mapsto\; x \;;_M \; y$$

*is called* `admissible (for modelling sequential composition)` *iff it satisfies:*

$$(x \;;_M \; y)[n] \;=\; \begin{cases} x[n] & : \; \text{if } \rho(x) \geq n \\ \\ x \;;_M \; y[n-m] & : \; \text{if } \rho(x) = m < n. \end{cases}$$

*for all $x$, $y \in M$ and $n \geq 1$.*

The following condition about the semantic operators $\omega_M$ is needed to get non-distance-increasing operators on the powerdomain. It asserts that the first $n$ steps of the possible computations of a composed program $\omega(P_1, \ldots, P_k)$ are uniquely determined by the first $n$ steps of the computations of $P_1, \ldots, P_k$. This requirements seems to be natural for operators like parallelism (with or without communication), hiding, relabelling or prefixing.

**Definition 3.4** *Let $M$ be a metric space which is equipped with a ranking and let $\omega \in \Omega$ be a $k$-ary operator symbol. An operator*

$$M^k \to \mathcal{P}_{\text{fin}}(M), \;\; (x_1, \ldots, x_k) \;\mapsto\; \omega_M(x_1, \ldots, x_k)$$

*is called* `admissible (for interpreting` $\omega$`)` *iff for all $\tilde{x} \in M^k$ and $n \geq 1$:*

$$\omega_M(\tilde{x}[n])[n] \;=\; \omega_M(\tilde{x})[n]$$

**Notation 3.5** *We say that a metric space $M$ is* `suitable to model finite behaviour` *iff it is equipped with a ranking and semantic operators $;_M$ and $\omega_M$, $\omega \in \Omega$, such that $;_M$ and the operators $\omega_M$ are admissible. In addition we suppose an interpretation of the atomic actions $\alpha \in Act$ in $M$, i.e. we assume that there are fixed elements $\alpha_M \in M$.*

**Example 3.6** It is easy to see that the sequence resp. parallel operators on $Act^+$ and $Pom^+$ satisfy the condition of Definition 3.3 resp. 3.4. Hence $Act^+$ and $Pom^+$ are suitable to model finite behaviour. □

## 3.2 Modelling infinite behaviour

In what follows we assume that $M$ is a metric space suitable to model finite behaviour. Let $\overline{M}$ be the completion of $M$. The elements of $\overline{M} \setminus M$ are considered as infinite computations. The underlying metric on $M$ and $\overline{M}$ is denoted by $d$. We extend the functions $M \to M$, $x \mapsto x[n]$, to functions

$$\overline{M} \to \overline{M}, \quad x \mapsto x[n]$$

as follows: If $x \in \overline{M}$ then $x[n] \in M$ denotes the unique element in $M$ such that for each Cauchy sequence $(x_m)$ in $M$ with $x = \lim x_m$:

$$x_m[n] = x[n]$$

for almost all $m \in \mathbb{N}_0$. Such an element $x[n]$ exists since for each two Cauchy sequences $(x_m)$, $(y_m)$ in $M$ with $x = \lim x_m = \lim y_m$ there exists $n_0 \geq 0$ such that $d(x_m, x)$, $d(y_m, x) \leq 1/2^n$ for all $m \geq n_0$. Then $d(x_m, y_m) \leq 1/2^n$. Hence $x_m[n] = y_m[n]$ for all $m \geq n_0$. It is easy to see that

$$d(x,y) = \inf \left\{ \frac{1}{2^n} : x[n] = y[n] \right\}$$

for all $x$, $y \in \overline{M}$. We extend $\rho$ to a function on $\overline{M}$:

$$\rho(x) = \infty \quad \text{if } x \in \overline{M} \setminus M$$

If $H \subseteq \overline{M}$ and $\tilde{x} = (x_1, \ldots, x_k) \in \overline{M}^k$ then we put:

$$H[n] = \{ x[n] : x \in H \}, \quad \tilde{x}[n] = (x_1[n], \ldots, x_k[n])$$

We also write $\tilde{H}[n]$ as an abbreviation for $H_1[n] \times \ldots \times H_k[n]$ if $\tilde{H} = H_1 \times \ldots \times H_k$.

**Example 3.7** The completion of $Act^+$ is $Act^\infty$ the set of nonempty, finite and infinite sequences over $Act$. The $n$-cut $x[n]$ of an infinite sequence is the $n$-th prefix. The completion of $Pom^+$ is the set $Pom^\infty$ of nonempty pomsets $p = (E, \leq, l)$ such that for each $n \geq 1$ the set $E[n]$ of all events $e \in E$ with $depth_p(e) \leq n$ is finite. The $n$-cut $p[n]$ of $p$ is the pomset which arises from $p$ by removing all events $e \notin E[n]$. □

**Definition 3.8** *The operator* $\overline{M} \times \overline{M} \to \overline{M}$, $(x,y) \mapsto x ;_{\overline{M}} y$ *is given by:*

$$x ;_{\overline{M}} y = \lim_{n \to \infty} x[n] ;_M y[n]$$

$;_{\overline{M}}$ is the canonical extension of $;_M$, i.e. $x ;_{\overline{M}} y = x ;_M y$ for all $x, y \in M$.

**Lemma 3.9** *For all* $x, y \in \overline{M}$ *and* $n \geq 1$*:*

$$(x ;_{\overline{M}} y)[n] = \begin{cases} x[n] & : \text{ if } \rho(x) \geq n \\ x ;_{\overline{M}} y[n-m] & : \text{ if } \rho(x) = m < n \end{cases}$$

*If* $x \in \overline{M} \setminus M$ *then* $x ;_{\overline{M}} y = x$.

**Proof**: easy verification. □

By Lemma 3.9 it follows immediately:

**Corollar 3.10** *For all $x$, $y \in \overline{M}$ and $n \geq 2$:*

$$(x \;;_{\overline{M}}\; y)[n] \;=\; (x[n] \;;_M\; y[n-1])[n]$$

*In particular, $;_{\overline{M}}$ is non-distance-increasing and contracting in its second argument.*

We extend the operators $\omega_M$ to operators $\omega_{\overline{M}}$ in the following way:

**Definition 3.11** *For each $k$-ary operator symbol $\omega \in \Omega$ we define an operator*

$$\omega_{\overline{M}} \;:\; \overline{M}^k \;\to\; \mathcal{P}(\overline{M})$$

*as follows:*

$$\omega_{\overline{M}}(\tilde{x}) \;=\; \left\{ \lim_{n\to\infty} z_n \;:\; z_n \in \omega_M(\tilde{x}[n])[n],\; z_n = z_{n+1}[n] \right\}$$

*Here $\mathcal{P}(\overline{M})$ stands for the powerset of $\overline{M}$.*

**Lemma 3.12** *$\omega_{\overline{M}}$ extends $\omega_M$, i.e. $\omega_M(\tilde{x}) = \omega_{\overline{M}}(\tilde{x})$ for all $\tilde{x} \in M^k$.*

**Proof**: Let $\tilde{x} \in M^k$. There exists a natural number $N \geq 1$ with $z[n] = z$ for all $z \in \omega_M(\tilde{x})$ and $n \geq N$. Note that the set $\omega_M(\tilde{x})$ is finite. Hence we may define

$$N \;=\; \max \{\; \rho(z) \;:\; z \in \omega_M(\tilde{x}) \;\}.$$

Then for all $n \geq N$:

$$\omega_M(\tilde{x}[n])[n] \;=\; \omega_M(\tilde{x})[n] \;=\; \omega_M(\tilde{x})$$

I.e. whenever $(z_n)$ is a sequence in $M$ with $z_n \in \omega_M(\tilde{x}[n])[n]$ and $z_n = z_{n+1}[n]$ then

$$z_N \;=\; z_{N+1}[N] \;=\; z_{N+1} \;=\; z_{N+2}[N+1] \;=\; z_{N+2} \;=\; \ldots$$

Therefore $\lim z_n \;=\; z_N \;\in\; \omega_M(\tilde{x})$. Hence $\omega_{\overline{M}}(\tilde{x}) \;\subseteq\; \omega_M(\tilde{x})$.

If $z \in \omega_M(\tilde{x})$ then we put $z_n = z[n]$. Then

$$z_n \;\in\; \omega_M(\tilde{x})[n] \;=\; \omega_M(\tilde{x}[n])[n],$$

$z_{n+1}[n] = z_n$ and $\lim z_n \;=\; z$. Hence $z \in \omega_{\overline{M}}(\tilde{x})$. □

**Lemma 3.13** *For each $k$-ary operator symbol $\omega \in \Omega$ and all $\tilde{x} \in \overline{M}^k$, $n \geq 1$:*

$$\omega_{\overline{M}}(\tilde{x})[n] \;=\; \omega_{\overline{M}}(\tilde{x}[n])[n]$$

*is a finite set.*

**Proof**: Let $z \in \omega_{\overline{M}}(\tilde{x})[n]$. Then there exists a sequence $(z_n)_{n\geq 1}$ such that

$$z_m \in \omega_M(\tilde{x}[m])[m]$$

and $z_m = z_{m+1}[m]$ and $z = (\lim z_m)[n]$. Then $z_n = z_m[n]$ for all $m \geq n$. By definition of the $n$-cuts of elements in $\overline{M}$ we have:

$$z = (\lim z_m)[n] = z_n \in \omega_{\overline{M}}(\tilde{x}[n])[n]$$

Let $z \in \omega_{\overline{M}}(\tilde{x}[n])[n]$. By Lemma 3.12:

$$z \in \omega_{\overline{M}}(\tilde{x}[n])[n] = \omega_M(\tilde{x}[n])[n]$$

We define by induction on $m \geq 1$ a sequence $(z_m)$ with $z_m \in \omega_M(\tilde{x}[m])[m]$, $z_m = z_m[m+1]$ and $z = (\lim z_m)[n]$.

- In the case $m \leq n$ we put: $z_m \stackrel{\text{def}}{=} z[m]$. Then:

$$z_m \in (\omega_M(\tilde{x}[n])[n])[m] = \omega_M(\tilde{x}[n])[m] = \omega_M(\tilde{x}[m])[m]$$

- We assume that $m \geq n$ and that $z_1, \ldots, z_m$ are defined. Since

$$z_m \in \omega_M(\tilde{x}[m])[m] = \omega_M(\tilde{x}[m+1])[m]$$

there exists $z'_{m+1} \in \omega_M(\tilde{x}[m+1])$ with $z'_{m+1}[m] = z_m$. We put:

$$z_{m+1} \stackrel{\text{def}}{=} z'_{m+1}[m+1]$$

Let $z' = \lim z_m$. Then $z' \in \omega_{\overline{M}}(\tilde{x})$ and $z = z'[n] \in \omega_{\overline{M}}(\tilde{x})[n]$.

By Lemma 3.12 and by the assumption that $\omega_M(\tilde{y})$ is finite for all $\tilde{y} \in M^k$:

$$\omega_{\overline{M}}(\tilde{x})[n] = \omega_M(\tilde{x}[n])[n]$$

is finite. □

## 3.3 A denotational linear time semantics on $\mathcal{P}_{\text{co}}(\overline{M})$

Our aim is to give a denotational semantics $Me$ for $Prog$ on some powerdomain construction of $\overline{M}$ such that $Me(P)$ can be viewed as the set of maximal computations of $P$. By our results in [1] we need a suitable powerdomain $\mathcal{P}_*(\overline{M})$ which is a complete metric space and which is endowed with semantic operators as follows:

- for each atomic action $\alpha \in Act$ there is an element $\overline{\alpha} \in \mathcal{P}_*(\overline{M})$
- there is a binary non-distance-increasing operator $\overline{;}$ on $\mathcal{P}_*(\overline{M})$ which is contracting in its second argument

- for each $k$-ary operator symbol $\omega \in \Omega$ there is a $k$-ary non-distance-increasing operator $\overline{\omega}$ on $\mathcal{P}_*(\overline{M})$
- there is a binary non-distance-increasing operator $\overline{+}$ on $\mathcal{P}_*(\overline{M})$

As shown in [1]: Under the assumptions of above there is a unique meaning function

$$Me^{\mathrm{cms}} \; : \; Prog \; \to \; \mathcal{P}_*(\overline{M})$$

which satisfies:

- $Me^{\mathrm{cms}}(\alpha) \; = \; \overline{\alpha}$
- $Me^{\mathrm{cms}}(\xi) \; = \; Me^{\mathrm{cms}}(\sigma(\xi))$
- $Me^{\mathrm{cms}}(P_1; P_2) \; = \; Me^{\mathrm{cms}}(P_1) \,\overline{;}\, Me^{\mathrm{cms}}(P_2)$
- $Me^{\mathrm{cms}}(P_1 + P_2) \; = \; Me^{\mathrm{cms}}(P_1) \,\overline{+}\, Me^{\mathrm{cms}}(P_2)$
- $Me^{\mathrm{cms}}(\omega(P_1, \ldots, P_k)) \; = \; \overline{\omega}(Me^{\mathrm{cms}}(P_1), \ldots, Me^{\mathrm{cms}}(P_k))$

It might be the case that there are several possibilities to define a domain $\mathcal{P}_*(\overline{M})$ which satisfies these properties. In order to guarantee an interpretation in terms of maximal computations we make some additional assumptions:

(I) For each atomic action $\alpha$ the associated meaning $\overline{\alpha}$ is the single-element set $\{\alpha_M\}$. This reflects the fact that the only computation of the program $P = \alpha$ is the computation which performs $\alpha$ and then stops.

(II) The semantic operator for modelling nondeterministic choice should be the union. This corresponds to the assumption that the maximal computations of $P + Q$ are exactly those of $P$ and $Q$.

(III) For the sequence operator $\overline{;}$ we require $H \,\overline{;}\, I \; = \; \{ \; x \;;_{\overline{M}} \; y \; : \; x \in H, \; y \in I \; \}$. This guarantees that $z \in Me^{\mathrm{cms}}(P; Q)$ if and only if either $z$ is an infinite computation of $P$ or $z$ represents a computation which starts by a terminating computation of $P$ followed by a computation of $Q$.

(IV) For each $k$-ary operator symbol $\omega$:

$$\overline{\omega}(H_1, \ldots, H_k) \; = \; \bigcup \{ \; \omega_{\overline{M}}(x_1, \ldots, x_k) \; : \; x_i \in H_i, \; i = 1, \ldots, k \; \}$$

This asserts that the maximal computations of the composed program $\omega(P_1, \ldots, P_k)$ are those which one gets by composing maximal computations of $P_1, \ldots, P_k$.

[10] and [12] have shown that the collection of all closed resp. compact subsets of a complete metric space endowed with the Hausdorff-distance

$$d(H, I) \; = \; \max \left\{ \sup_{x \in H} d(x, I), \sup_{y \in I} d(y, H) \right\} \quad \text{where} \quad d(z, X) \; = \; \inf_{x \in X} d(x, z)$$

are complete metric spaces. Hence we have two candidates as semantic domain:

$\mathcal{P}_{\mathrm{cl}}(\overline{M})$ the collection of all nonempty and closed subsets of $M$

$\mathcal{P}_{\mathrm{co}}(\overline{M})$ the collection of all nonempty and compact subsets of $M$

The reason for excluding the empty set is that the empty set has no interpretation as a computation (and would cause problems to define a sequential operator which is contracting in its second argument). In both cases the single-element sets $\{\alpha_M\}$ are suitable interpretations of the atomic actions $\alpha \in Act$. The use of the union for modelling nondeterministic choice causes no problem since $\mathcal{P}_{\mathrm{cl}}(\overline{M})$ and $\mathcal{P}_{\mathrm{co}}(\overline{M})$ are closed under union and since the union is always non-distance-increasing w.r.t. the Hausdorff-distance. Defining suitable semantic operators on $\mathcal{P}_{\mathrm{cl}}(\overline{M})$ might be problematic in the case where $\overline{M}$ is not compact. This is because there might be sets $H$, $I \in \mathcal{P}_{\mathrm{cl}}(\overline{M})$ such that the set $\{ x ;_{\overline{M}} y : x \in H, y \in I \}$ is not closed. Warmerdam illustrated this by an example in the case $M = Act^+$ which can be found in [6]. Similary it might be the case that for closed sets $H_1, \ldots, H_k$ the set

$$\bigcup \{ \omega_{\overline{M}}(x_1, \ldots, x_k) : x_i \in H_i,\ i = 1, \ldots, k \}$$

is not closed. If one is forced to use $\mathcal{P}_{\mathrm{cl}}(\overline{M})$ as semantic domain (e.g. if one deals infinite nondeterminism instead of our binary choice operator $+$) then using the closure of those sets yield semantic operators on $\mathcal{P}_{\mathrm{cl}}(\overline{M})$ which satisfies all mathematical properties which are needed to define a denotational semantics on $\mathcal{P}_{\mathrm{cl}}(\overline{M})$. We argue that the resulting semantics is not adequate because there it might include infinite elements in the meaning of a program $P$ which do not represent a possible computation of $P$.

Since our language *Prog* does not allow for infinite nondeterminism we may deal with the powerdomain construction $\mathcal{P}_{\mathrm{co}}(\overline{M})$. In the rest of this section we show how a denotational linear time semantics on $\mathcal{P}_{\mathrm{co}}(\overline{M})$ can be defined. It is easy to see that for all $H$, $I \in \mathcal{P}_{\mathrm{co}}(\overline{M})$:

$$d(H, I) = \inf \left\{ \frac{1}{2^n} : H[n] = I[n] \right\}.$$

I.e. the metric on $\mathcal{P}_{\mathrm{co}}(\overline{M})$ is induced by the ranking $H \mapsto H[n]$. Hence $d(H, I) \leq 1/2^n$ if and only if for each computation $x \in H$ there is some $y \in I$ with $x[n] = y[n]$ and vice versa. Before we define the semantic operators on $\mathcal{P}_{\mathrm{co}}(\overline{M})$ we give a characterization of the elements of $\mathcal{P}_{\mathrm{co}}(\overline{M})$:

**Lemma 3.14** *Let $H$ be a closed subset of $\overline{M}$. Then $H$ is compact if and only if for each $n \geq 1$ the set $H[n]$ is finite.*

**Proof**: If $H$ is compact and $n \geq 1$ then the open balls $B(x, 1/2^{n-1})$ with center $x \in H$ and radius $1/2^{n-1}$ form an open cover of $H$. Hence there exists a finite subcovering. I.e. there exist $x_1, \ldots, x_k \in H$ such that each element $x \in H$ is contained in a ball $B(x_i, 1/2^{n-1})$. Hence $d(x, x_i) < 1/2^{n-1}$. Therefore $d(x, x_i) \leq 1/2^n$, i.e. $x[n] = x_i[n]$. We conclude that $H[n] = \{x_1[n], \ldots, x_k[n]\}$ is finite.

Now we assume that the sets $H[n]$ are finite. We show that each sequence $(x_k)$ in $H$ contains a convergent subsequence $(x_{k_n})$ whose limit belongs to $H$. We define the subsequence

$(x_{k_n})$ and infinite sets $I_n$ of natural numbers by induction on $n$ such that $x_{k_n}[n] = x_k[n]$ for all $k \in I_n$. Then $(x_{k_n})$ is a Cauchy sequence in $\overline{M}$. Hence $\lim x_{k_n}$ exists. Since $H$ is closed $\lim x_{k_n} \in H$.

Since $H[1]$ is finite and since $x_k[1] \in H[1]$ for all $k \geq 1$ there is an infinite set $I_1$ of indices $k \geq 1$ and some $k_1 \geq 1$ such that $x_{k_1}[1] = x_k[1]$ for all $k \in I_1$.

Now we assume that $n \geq 2$ and $x_{k_1}, \ldots, x_{k_{n-1}}$ and $I_{n-1}$ are defined. Since $H[n]$ is finite and since $x_k[n] \in H[n]$ for all $k \in I_{n-1}$ there exists an infinite subset $I_n$ of $I_{n-1}$ and an element $k_n \in I_n$ with $k_n > k_{n-1}$ such that $x_{k_n}[n] = x_k[n]$ for all $k \in I_n$. □

**Lemma 3.15** *If $H$, $I \in \mathcal{P}_{\mathrm{co}}(\overline{M})$ then the set*

$$H \,\tilde{;}\, I \stackrel{\text{def}}{=} \{\, x \,;_{\overline{M}}\, y \;:\; x \in H,\ y \in I \,\}$$

*is compact and satisfies $(H \,\tilde{;}\, I)[n] = (\, H[n] \,\tilde{;}\, I[n-1] \,)[n]$ for all $n \geq 2$.*

**Proof**: The formula $(H \,\tilde{;}\, I)[n] = (\, H[n] \,\tilde{;}\, I[n-1] \,)[n]$ follows immediately by Corollar 3.10. By Lemma 3.14: If $H$, $I$ are compact then the sets $H[n]$ and $I[n-1]$ are finite. Hence $H[n] \,\tilde{;}\, I[n-1]$ and therefore $(H \,\tilde{;}\, I)[n]$ is finite. Now we show that $H \,\tilde{;}\, I$ is closed: Let $(z_n)$ be a sequence in $H \,\tilde{;}\, I$ which converges in $\overline{M}$ to $z$. We have to show that $z \in H \,\tilde{;}\, I$. Let $(x_n)$ resp. $(y_n)$ be sequences in $H$ resp. $I$ such that:

$$z_n = x_n \,;_{\overline{M}}\, y_n$$

Since $H$ is compact there is convergent subsequence $(x_{n_k})$. Then $\lim x_{n_k} \in H$. W.l.o.g. $x_{n_k} = x_k$. Otherwise we deal with the subsequence $(z_{n_k})$ instead of $(z_n)$. Since $I$ is compact there exists a convergent subsequence $(y_{n_k})$ of $(y_n)$. Then $\lim y_{n_k} \in I$. Then (since $;_{\overline{M}}$ is non-distance-increasing and therefore continuous):

$$\begin{aligned} z &= \lim_{k\to\infty} z_{n_k} = \lim_{k\to\infty} x_{n_k} \,;_{\overline{M}}\, y_{n_k} \\ &= \left(\lim_{k\to\infty} x_{n_k}\right) \,;_{\overline{M}}\, \left(\lim_{k\to\infty} y_{n_k}\right) \in H \,\tilde{;}\, I \end{aligned}$$

By Lemma 3.14: $H \,\tilde{;}\, I$ is compact. □

**Corollar 3.16** *The operator $\tilde{;} : \mathcal{P}_{\mathrm{co}}(\overline{M}) \times \mathcal{P}_{\mathrm{co}}(\overline{M}) \to \mathcal{P}_{\mathrm{co}}(\overline{M})$ is non-distance-increasing and contracting in its second argument.*

**Proof**: follows immediately by the equation $(H \,\tilde{;}\, I)[n] = (H[n] \,\tilde{;}\, I[n-1])[n]$. □

**Definition 3.17** *If $\omega \in \Omega$ is a $k$-ary operator symbol then we define:*

$$\overline{\omega}(H_1, \ldots, H_k) \stackrel{\text{def}}{=} \bigcup \{\, \omega_{\overline{M}}(x_1, \ldots, x_k) \;:\; x_i \in H_i,\ i = 1, \ldots, k \,\}$$

*for all $H_1, \ldots, H_k \in \mathcal{P}_{\mathrm{co}}(\overline{M})$.*

**Lemma 3.18** *If $H_1, \ldots, H_k$ are compact then also $\overline{\omega}(H_1, \ldots, H_k)$ is compact. For all $n \geq 1$ we have:*

$$\overline{\omega}(H_1, \ldots, H_k)[n] = \overline{\omega}(H_1[n], \ldots, H_k[n])[n]$$

**Proof**: The formula $\overline{\omega}(H_1,\ldots,H_k)[n] = \overline{\omega}(H_1[n],\ldots,H_k[n])[n]$ follows immediately by Lemma 3.13. If $H_i$ are compact, $i = 1,\ldots,k$, then $H_i[n]$ are finite sets (Lemma 3.14). Let $\tilde{H} = H_1 \times \ldots \times H_k$. Then $\tilde{H}$ is compact and $\tilde{H}[n]$ finite. Hence the sets

$$\overline{\omega}(\tilde{H}[n])[n] = \bigcup \{ \omega_M(\tilde{x}[n])[n] : \tilde{x} \in \tilde{H} \}$$

are finite. (Here we use Lemma 3.12.) Therefore the sets $\overline{\omega}(\tilde{H})[n]$, $n \geq 1$, are finite.

Now we show that the set $\overline{\omega}(\tilde{H})$ is closed. Let $(z_n)$ be a convergent sequence in $\overline{\omega}(\tilde{H})$ and $z = \lim z_n$. We have to show that $z \in \overline{\omega}(\tilde{H})$. W.l.o.g. $d(z_n, z) \leq 1/2^n$. Then

$$z[n] = z_n[n] \in \overline{\omega}(\tilde{H})[n] = \overline{\omega}(\tilde{H}[n])[n]$$

Hence there exists a sequence $(\tilde{x}_n)$ in $\tilde{H}$ with $z[n] \in \omega_M(\tilde{x}_n[n])[n]$. Since $\tilde{H}$ is compact there exists a convergent subsequence of $(\tilde{x}_n)$. Let $\tilde{x}$ be the limit of this sequence. We show that $z[n] \in \omega_M(\tilde{x}[n])[n]$. If $n \geq 1$ then $\tilde{x}[n] = \tilde{x}_m[n]$ for some $m \geq n$. Hence

$$\omega_M(\tilde{x}[n])[n] = \omega_M(\tilde{x}_m[n])[n] = \omega_M(\tilde{x}_m[m])[n]$$

Since $z[m] \in \omega_M(\tilde{x}_m[m])[m]$ we get:

$$z[n] = (z[m])[n] \in \omega_M(\tilde{x}_m[m])[n] = \omega_M(\tilde{x}[n])[n]$$

Since $\tilde{H}$ is closed $\tilde{x} \in \tilde{H}$. We conclude:

$$z = \lim_{n\to\infty} z[n] \in \overline{\omega}(\tilde{H})$$

By Lemma 3.14 $\overline{\omega}(\tilde{H})$ is compact. □

**Corollar 3.19** *For each $k$-ary operator symbol $\omega \in \Omega$ the operator*

$$\overline{\omega} : \mathcal{P}_{\mathrm{co}}(\overline{M})^k \to \mathcal{P}_{\mathrm{co}}(\overline{M})$$

*is welldefined and non-distance-increasing.*

Using the semantic operators $\bar{;}$, $\overline{\omega}$ and the union for modelling nondeterminism we get a denotational linear time semantics:

$$Me^{\mathrm{cms}} : Prog^{\sigma}(\Omega) \to \mathcal{P}_{\mathrm{co}}(\overline{M})$$

where the action symbols $\alpha \in Act$ are interpreted by $\overline{\alpha} \stackrel{\mathrm{def}}{=} \{ \alpha_M \}$.

**Example 3.20** The trace semantics of the program $P = \xi + \beta$ where $\sigma(\xi) = \alpha;\xi$ is $\{ \alpha\alpha\alpha\ldots, \beta \}$. The trace semantics of $Q = \gamma_1 \parallel \gamma_2$ is $\{ \gamma_1\gamma_2, \gamma_2\gamma_1 \}$. Hence the trace semantics of $P;Q$ is the set of traces which we get by applying the operator $\bar{;}$ to the trace semantics of $P$ and $Q$:

$$\{ \alpha\alpha\alpha\ldots, \beta\gamma_1\gamma_2, \beta\gamma_2\gamma_1 \}$$

The pomset semantics of $P$ consists of the infinite pomset

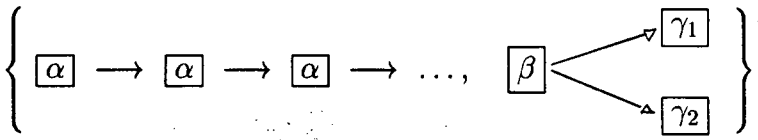
$$\boxed{\alpha} \longrightarrow \boxed{\alpha} \longrightarrow \boxed{\alpha} \longrightarrow \ldots$$

and the pomset $\boxed{\beta}$. The pomset semantics of $Q$ is the single-element set consisting of the pomset

$\boxed{\gamma_1}$

$\boxed{\gamma_2}$

The pomset semantics of $P;Q$ arises from the semantics of $P$ and $Q$ by applying the operator $\bar{;}$:



□

# 4 Denotational linear time semantics in the partial order approach

In this section we discuss the use of the partial order approach to give denotational linear time semantics for the language *Prog*. First we argue that maximal computations cannot be expressed by the partial order approach. Second we present conditions which allow the definition of a denotational linear time semantics which assigns to each program the set of its partial computations.

We claim that there do not exist

(1) a semantic domain $A$ whose elements can be interpreted as maximal computations

(2) a powerdomain construction $\mathcal{P}_*(A)$ of $A$ which is endowed with a partial order $\sqsubseteq$ such that $\mathcal{P}_*(A)$ is a cpo and the union is monotone on $\mathcal{P}_*(A)$

such that for each declaration $\sigma$ a meaning function $Me : Prog^{\sigma}(\Omega) \to \mathcal{P}_*(A)$ can be defined which satisfies:

(3) $Me(P)$ is the set of maximal computations of $P$.

(4) For each recursive program: the sequence of its finite approximations is monotone and its meaning is the least upper bound of its finite approximations.

We do not give formal descriptions of the assumptions (1), (3) and (4). Condition (4) (as we use it in the proof) is satisfied when $Me$ is a denotational semantics defined by the standard procedure: using continuous semantic operators and Tarski's fixed point thoerem.

**Proof**: Let $\sigma$ be a declaration such that

$$\begin{aligned} \sigma(\zeta) &= \alpha;\zeta \\ \sigma(\xi) &= \alpha;\alpha;\xi + \alpha;\alpha \\ \sigma(\eta) &= \alpha;\alpha;(\eta + \alpha) + \alpha;\alpha \end{aligned}$$

We assume that a meaning function $Me : Prog \to \mathcal{P}_*(A)$ satisfying the conditions (1)-(4) exists (where the underlying declaration of $Prog$ is $\sigma$). We put:

$$\begin{array}{lcllcl} S_1 & = & \alpha & S_{n+1} & = & \alpha; S_n \\ P_1 & = & \alpha;\alpha \ + \ \alpha;\alpha & P_{n+1} & = & \alpha;\alpha;P_n \ + \ \alpha;\alpha \\ Q_1 & = & \alpha;\alpha;\alpha \ + \ \alpha;\alpha & Q_{n+1} & = & \alpha;\alpha;(Q_n + \alpha) \ + \ \alpha;\alpha \end{array}$$

We assume that (1) guarantees the existence of pairwise distinct elements $x_n$, $n \geq 1$, and $x \in A$ such that $x_n$ represents the computations which performs $n$-times the action $\alpha$ and then stops and $x$ a computations which performs the action $\alpha$ infinitely often. Then by condition (3):

$$\begin{array}{lcl} Me(S_n) & = & \{x_n\} \\ Me(P_n) & = & \{\ x_{2k} \ : \ 1 \leq k \leq n\ \} \\ Me(Q_n) & = & \{\ x_k \ : \ 2 \leq k \leq 2n+1\ \} \end{array}$$

$S_n$, $P_n$ resp. $Q_n$ are the $n$-th unwindings of the recursive programs $\zeta$, $\xi$ resp. $\eta$. Hence $Me(S_n)$, $Me(P_n)$ resp. $Me(Q_n)$ are considered as the $n$-th approximation of $Me(\zeta)$, $Me(\xi)$ resp. $Me(\eta)$. By assumption (4):

(i) $Me(\zeta) = \bigsqcup Me(S_n)$

(ii) $Me(\xi) = \bigsqcup Me(P_n)$

(iii) $Me(\eta) = \bigsqcup Me(Q_n)$

On the other hand by assumption (3):

(i') $Me(\zeta) = \{x\}$

(ii') $Me(\xi) = \{\ x_{2k} \ : \ k \geq 1\ \} \cup \{x\}$

(iii') $Me(\eta) = \{\ x_k \ : \ k \geq 1\ \} \cup \{x\}$

By (i) and (i'):

(iv) $\{x_1\} \sqsubseteq \{x_2\} \sqsubseteq \{x_3\} \sqsubseteq \ldots \sqsubseteq \{x\}$

By the monotonicity of the union (assumption (2)) and by (iv):

$$\begin{array}{rcl} Me(P_n) & = & \{x_2\} \cup \{x_4\} \cup \ldots \cup \{x_{2n}\} \\ & = & \{x_2\} \cup \{x_2\} \cup \{x_4\} \cup \{x_4\} \cup \ldots \cup \{x_{2n}\} \cup \{x_{2n}\} \\ & \sqsubseteq & \{x_2\} \cup \{x_3\} \cup \{x_4\} \cup \{x_5\} \cup \ldots \cup \{x_{2n}\} \cup \{x_{2n+1}\} \\ & = & Me(Q_n) \end{array}$$

Hence by (ii) and (iii) $Me(\xi) \sqsubseteq Me(\eta)$. Again by the monotonicity of the union (assumption (2)) and by (iv):

$$\begin{aligned} Me(Q_n) &= \{x_2\} \cup \{x_3\} \cup \{x_4\} \cup \ldots \cup \{x_{2n-1}\} \cup \{x_{2n}\} \cup \{x_{2n+1}\} \\ &\sqsubseteq \{x_2\} \cup \{x_4\} \cup \{x_4\} \cup \ldots \cup \{x_{2n}\} \cup \{x_{2n}\} \cup \{x_{2n+2}\} \\ &= \{x_2\} \cup \{x_4\} \cup \ldots \cup \{x_{2n}\} \cup \{x_{2n+2}\} \\ &= Me(P_{n+1}) \end{aligned}$$

Hence by (ii) and (iii) $Me(\eta) \sqsubseteq Me(\xi)$. Therefore: $Me(\eta) = Me(\xi)$. On the other hand by (ii') and (iii'):

$$x_3 \in Me(\eta) \setminus Me(\xi)$$

Contradiction. □

The result of above carries over to languages like $CCS$ which use a prefix operator $P \mapsto \alpha.P$ instead of sequential composition. If programs are given by a production system of the form

$$P ::= nil \mid \xi \mid P_1 + P_2 \mid \alpha.P \mid \ldots$$

where a fixed declaration $\sigma$ is assumed and where *nil* stands for a process which does not perform any action then in the above we only have to modify the syntax in which the programs $S_n$, $P_n$, $Q_n$ and the statements $\sigma(\cdot)$ are defined. E.g. $\sigma(\xi) = \alpha.\alpha.\xi + \alpha.\alpha.nil$.

## 4.1 Pointed posets suitable to model finite behaviour

We show how a denotational linear time semantics in the partial order approach can be defined such that the meaning of a program is the set of its partial computations. The starting point is a pointed poset $D$, i.e. a partially ordered set $D$ with a bottom element $\perp$, whose elements can be considered as computations of non-recursive programs. We suppose that $D$ does not distinguish computations leading to an intermediate state and their terminating counterparts. We assume that the underlying partial order $\sqsubseteq$ on $D$ can be interpreted as follows: $x \sqsubseteq y$ if and only if $x$ is a 'subcomputation' of $y$. The bottom element is assumed to represent a process that does not perform any action.

**Example 4.1** The prefix ordering on $Act^*$ and also the partial order

$$p \sqsubseteq p' \iff \begin{cases} \text{There are representatives } (E, \leq, l) \text{ resp. } (E', \leq', l') \text{ of} \\ p \text{ and } p' \text{ with } E \subseteq E', \leq = \leq' \cap E \times E \text{ and } l = l'|E. \end{cases}$$

on $Pom^*$ allow such an interpretation. In the case of traces $\perp$ is the empty trace (a sequence of length 0), in the case of pomsets $\perp$ is the empty pomset $(\emptyset, \emptyset, \emptyset)$. □

As in the metric case we formalize the conditions that a suitable sequence operator on $D$ has to fulfill such that our intuition is reflected: The bottom element should be neutral which corresponds to the fact that the process which does not perform any action is neutral w.r.t. sequential composition. Recall that we do not deal with deadlocked processes, hence

$\perp$ stands for a well-terminated process. Second we require the monotonicity in the second argument. This reflects the assumption that for deterministic terminating processes $P$, $P_1$, $P_2$: If the execution of $P_1$ is a partial computation of $P_2$ then the execution of $P; P_1$ is a partial computation of $P; P_2$. The third condition asserts that for deterministic terminating programs $P$, $P'$: $z$ is a partial computation of $P; P'$ if and only if $z$ is either a partial computation of $P$ or $z$ represents the execution of the whole program $P$ followed by a partial computation of $P'$.

**Definition 4.2** *An operator* $;_D : D \times D \to D$ *is called* `admissible (for modelling sequential composition)` *iff it satisfies:*

- $\perp ;_D x = x ;_D \perp = x$
- $;_D$ *is monotone in its second argument, i.e.* $y \sqsubseteq z$ *implies* $x ;_D y \sqsubseteq x ;_D z$
- $z \sqsubseteq x ;_D y$ *if and only if either* $z \sqsubseteq x$ *or there is some* $w \in D$ *such that* $w \sqsubseteq y$ *and* $z = x ;_D w$.

From the mathematical point of view there is no need to make any requirements about the operators $\omega_D$ on $D$. A natural assumption (which we make in section 5) would be the monotonicity of $\omega_D$ w.r.t. the lower preorder.

**Notation 4.3** *We say that a pointed poset* $D$ *is* `suitable to model finite behaviour` *iff* $D$ *is endowed with an admissible semantic operator* $;_D$ *and for each* $k$*-ary operator symbol* $\omega \in \Omega$ *an operator*

$$\omega_D : D^k \to \mathcal{P}_{\text{fin}}(D).$$

*In addition we require that the actomic actions* $\alpha \in Act$ *are interpreted by fixed elements* $\alpha_D \in D$.

**Example 4.4** The sequence and parallel operators on $Act^*$ and $Pom^*$ are admissible. Hence $Act^*$ and $Pom^*$ are pointed posets suitable for modelling finite behaviour. □

In the sequel we assume that $D$ is a pointed poset which is suitable to model finite behaviour. Our aim is to give a denotational linear time semantics for *Prog* on some powerdomain $\mathcal{P}_*(D)$ of $D$. By our results of [1] we need a powerdomain $\mathcal{P}_*(D)$ which is a cpo and which is endowed with semantic operators as follows:

- for each atomic action $\alpha \in Act$ there is an element $\hat{\alpha} \in \mathcal{P}_*(D)$
- there is a continuous operator $\hat{;} : \mathcal{P}_*(D) \times \mathcal{P}_*(D) \to \mathcal{P}_*(D)$
- for each $k$-ary operator symbol $\omega \in \Omega$ there is a $k$-ary continuous operator $\hat{\omega}$ on $\mathcal{P}_*(D)$
- there is a continuous operator $\hat{+} : \mathcal{P}_*(D) \times \mathcal{P}_*(D) \to \mathcal{P}_*(D)$

As shown in [1]: Under these assumptions there is a least meaning function

$$Me^{\text{cpo}} : Prog \to \mathcal{P}_*(D)$$

which satisfies:

- $Me^{\mathrm{cpo}}(\alpha) = \widehat{\alpha}$
- $Me^{\mathrm{cpo}}(\xi) = Me^{\mathrm{cpo}}(\sigma(\xi))$
- $Me^{\mathrm{cpo}}(P_1; P_2) = Me^{\mathrm{cpo}}(P_1) \,\widehat{;}\, Me^{\mathrm{cpo}}(P_2)$
- $Me^{\mathrm{cpo}}(P_1 + P_2) = Me^{\mathrm{cpo}}(P_1) \,\widehat{+}\, Me^{\mathrm{cpo}}(P_2)$
- $Me^{\mathrm{cpo}}(\omega(P_1, \ldots, P_k)) = \widehat{\omega}(Me^{\mathrm{cpo}}(P_1), \ldots, Me^{\mathrm{cpo}}(P_k))$

In order to ensure that the $Me^{\mathrm{cpo}}(P)$ can be considered as the set of partial computations of $P$ and because of our assumption that $x \sqsubseteq y$ if and only if $x$ is a 'subcomputation' of $y$ we get that the elements of $\mathcal{P}_*(D)$ are leftclosed (i.e. all predecessors of an element in $H \in \mathcal{P}_*(D)$ are contained in $H$). This is because whenever $y \in Me^{\mathrm{cpo}}(P)$ and $x \sqsubseteq y$ then $y$ (and then also $x$) is a partial computation of $P$. Hence $x \in Me^{\mathrm{cpo}}(P)$. I.e. $Me^{\mathrm{cpo}}(P)$ is leftclosed. Hence, powerdomain constructions like the Plotkin [14] or Smyth [16] powerdomain are not suitable to describe partial computations. We generalize the idea of [11] and extend $D$ by new elements such that computations leading to an intermediate state and their terminating counterparts can be distinguished and use the cpo of leftclosed subsets of this extension of $D$ as semantic domain.

## 4.2 Modelling partial behaviour

Following the idea of [11] we duplicate the elements of $D$ to get new elements $x\surd$ whose behaviour equals those of $x$ with the exception that we think of $x$ to describe a computation which does not come to a halt and and $x\surd$ to represent a terminating process. One might think of $\surd$ as a new action that a terminating process performs at the end of its execution and that informs the environment about its termination.

**Definition 4.5** *Let $D$ be a pointed poset. For each $x \in D$ let $x\surd$ be an element such that $x\surd \notin D$ and $x\surd \neq y\surd$ if $x \neq y$. Then we define:*

$$D_\surd = D \cup \{x\surd : x \in D\}$$

*The partial order $\sqsubseteq_\surd$ on $D_\surd$ is given by:*

*(1) If $x, y \in D$ then $x \sqsubseteq_\surd y$ if and only if $x \sqsubseteq y$.*

*(2) If $x \in D_\surd$, $y \in D$ then $x \sqsubseteq_\surd y\surd$ if and only if $x \in D$ and $x \sqsubseteq y$.*

*(3) If $x \in D$, $y \in D_\surd$ then $x\surd \sqsubseteq_\surd y$ if and only if $x\surd = y$.*

$\sqsubseteq_\surd$ is the smallest partial order on $D_\surd$ such that $\sqsubseteq = \sqsubseteq_\surd \cap D \times D$ and $x \sqsubseteq_\surd x\surd$. Hence we may write $\sqsubseteq$ instead of $\sqsubseteq_\surd$. The elements $x\surd$ are maximal in $D_\surd$.

**Notation 4.6** *The projection $\pi : D_\surd \to D$ is given by:*

$$\pi(x) = \begin{cases} x & : \text{if } x \in D \\ x' & : \text{if } x = x'\surd \end{cases}$$

**Example 4.7** The associated poset $Act^*_{\surd}$ can be identified with the set of finite sequences over $Act \cup \{\surd\}$ which either do not contain $\surd$ or which contain $\surd$ only as last element. If $x = \alpha_1 \ldots \alpha_n \in Act^*$ then $x\surd = \alpha_1 \ldots \alpha_n\surd$. Then $\pi(x) = x$ for each sequence $x$ over $Act$. If $x$ ends with the element $\surd$ then $\pi(x)$ is the sequence which arises from $x$ by removing $\surd$.

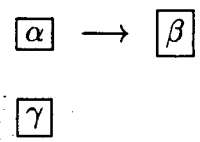
The associated poset $Pom^*_{\surd}$ can be identified with the set of finite pomsets $(E, \leq, l)$ where the labelling function $l$ maps the events to the set $Act \cup \{\surd\}$ and where an event $e \in E$ is allowed to have the action $\surd$ only if $e$ is the greatest element of $(E, \leq)$. If $p = (E, \leq, l)$ is a finite pomset then

$$p\surd = (E \cup \{e_0\}, \leq', l')$$

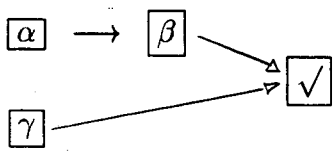
where $e_0 \notin E$, $l'(e) = l(e)$ if $e \in E$ and $l'(e_0) = \surd$,

$$e \leq e' \iff e \leq e' \vee e' = e_0.$$

I.e. $p\surd$ arises from $p$ by appending an event labelled by $\surd$. For instance, if $p$ is given by




then $p\surd$ is given by




If $p$ is a pomsets whose events are all labelled by actions $\alpha \in Act$ then $\pi(p) = p$. Otherwise $\pi(p)$ is the pomset which we get by removing the event which is labelled by $\surd$. □

The operator $;_D$ on $D$ induces an operator $;_{\surd}$ on $D_{\surd}$ as follows: If $x\surd$ is a computation of $P$ and $y$ a computation of some program $Q$ then $x$ followed by $y$ is a computation of $P;Q$. If $x \in D$ is a partial nonterminating computation of $P$ then $x$ is also a partial nonterminating computation of $P;Q$. This is reflected in the following definition:

**Definition 4.8** *If $x, y \in D_{\surd}$ we put:*

$$x ;_{\surd} y = \begin{cases} x' ;_D y & : \text{if } x = x'\surd,\ y \in D \\ (x' ;_D y')\surd & : \text{if } x = x'\surd,\ y = y'\surd \\ x & : \text{if } x \in D \end{cases}$$

In general an admissible operator $;_D$ is not monotone. For instance the sequence operator on $Act^*$ is not monotone: $\alpha \sqsubseteq \alpha\beta$ but $\alpha\gamma \not\sqsubseteq \alpha\beta\gamma$. Nevertheless the associated operator $;_{\surd}$ is monotone:

**Lemma 4.9** *The operator $;_{\surd}$ is monotone on $D_{\surd}$.*

**Proof**: Let $x \sqsubseteq x', y \sqsubseteq y'$.

Case 1: $x \in D$. If $x' \in D$ then $x ;_{\surd} y = x \sqsubseteq x' = x' ;_{\surd} y'$. Otherwise $x' = z\surd$ for some $z \in D$. Then $x \sqsubseteq z$ and

$$x' ;_{\surd} y' = \begin{cases} z ;_D y' & : \text{ if } y' \in D \\ (z ;_D w)\surd & : \text{ if } y' = w\surd. \end{cases}$$

In both cases $z \sqsubseteq x' ;_{\surd} y'$. Hence $x ;_{\surd} y = x \sqsubseteq z \sqsubseteq x' ;_{\surd} y'$.

Case 2: $x = z\surd$. Then $x$ is maximal. Hence $x = x'$. If $y \notin D$ then $y$ is maximal and hence $y' = y$. Therefore $x ;\surd y = x' ;_{\surd} y'$. If $y \in D$ then $y \sqsubseteq \pi(y')$ and

$$x ;_{\surd} y = z ;_D y \sqsubseteq z ;_D \pi(y') \sqsubseteq \begin{cases} z ;_D y' = x' ;_{\surd} y' & : \text{ if } y' \in D \\ z ;_D w \sqsubseteq x' ;_{\surd} y' & : \text{ if } y' = w\surd \end{cases}$$

□

Next we extend the operators $\omega_D$ on $D$ to operators $\omega_{\surd}$ on $D_{\surd}$. Here we assume that a computation $x \in \omega_D(\tilde{x})$ terminates if and only if $x_1, \ldots, x_k$ are terminating computations.

**Definition 4.10** *Let $\omega$ be a $k$-ary operator symbol in $\Omega$. Then $\omega_{\surd} : D_{\surd}^k \to \mathcal{P}_{\text{fin}}(D_{\surd})$ is given by:*

- *If $x_1, \ldots, x_k \in D_{\surd}$ where $x_i \in D$ for some $i$ then:*

$$\omega_{\surd}(x_1, \ldots, x_k) \stackrel{\text{def}}{=} \omega_D(\pi(x_1), \ldots, \pi(x_k))$$

- *If $x_1, \ldots, x_k \in D_{\surd} \setminus D$ then:*

$$\omega_{\surd}(x_1, \ldots, x_k) \stackrel{\text{def}}{=} \{\, z\surd \,:\, z \in \omega_D(\pi(x_1), \ldots, \pi(x_k)) \,\}$$

## 4.3 A denotational linear time semantics on $\mathcal{P}_{\surd}(D)$

We give a denotational linear time semantics on the cpo of leftclosed and nonempty subsets of $D_{\surd}$ such that the programs are descibed by their partial computations.

**Definition 4.11** *$\mathcal{P}_{\surd}(D)$ denotes the cpo of nonempty and leftclosed subsets of $D_{\surd}$, i.e.*

$$\mathcal{P}_{\surd}(D) = \mathcal{P}_{\downarrow}(D_{\surd})$$

*ordered by inclusion.*

It is clear that $\mathcal{P}_{\surd}(D)$ is a cpo with bottom element $\{\bot\}$. The least upper bound of a (monotone) sequence $(H_n)$ in $\mathcal{P}_{\surd}(D)$ is $\bigcup H_n$. We show that the operators on $D_{\surd}$ induces continuous operators on $\mathcal{P}_{\surd}(D)$.

**Definition 4.12** *If $H, I \in \mathcal{P}_\surd(D)$ then we put:*

$$H \mathbin{\hat{;}} I \stackrel{\text{def}}{=} \{ x \mathbin{\hat{;}_\surd} y : x \in H, y \in I \}$$

**Lemma 4.13** *If $H$, $I$ are leftclosed then $H \mathbin{\hat{;}} I$ is leftclosed.*

**Proof**: Let $x \in H$, $y \in I$ and $z \sqsubseteq x ;_\surd y$. If $z \in D_\surd \setminus D$ then $z$ is maximal and therefore

$$z = x ;_\surd y \in H \mathbin{\hat{;}} I.$$

Now we assume that $z \in D$. Then $z \sqsubseteq x ;_\surd \pi(y)$. Since $\pi(y) \in I$ we may assume that $y = \pi(y) \in D$.

If $x \in D$ then $x ;_\surd y = x$ and $z \sqsubseteq x$. Then $z \in H \cap D$ and therefore

$$z = z ;_\surd y \in H \mathbin{\hat{;}} I.$$

If $x = x'\surd$ then $x ;_\surd y = x' ;_D y$. Hence either $z \sqsubseteq x$ and therefore $z \in H \cap D$ or $z = x' ;_D w$ for some $w \sqsubseteq y$. In the first case:

$$z = z ;_\surd y \in H \mathbin{\hat{;}} I.$$

In the second case $w \in I$ and $z = x ;_\surd w \in H \mathbin{\hat{;}} I$. □

Immediately by the definition of $\mathbin{\hat{;}}$ we get:

**Lemma 4.14** *The operator $\mathbin{\hat{;}} : \mathcal{P}_\surd(D) \times \mathcal{P}_\surd(D) \to \mathcal{P}_\surd(D)$ is continuous.*

**Definition 4.15** *If $\omega$ is a $k$-ary operator symbol we put:*

$$\hat{\omega}(\tilde{H}) \stackrel{\text{def}}{=} \bigcup \{ \omega_\surd(\tilde{x}) \downarrow : \tilde{x} \in \tilde{H} \}$$

*for all $\tilde{H} \in \mathcal{P}_\surd(D)^k$.*

Here $H \downarrow$ denotes the leftclosure of $H$, i.e.

$$H \downarrow \stackrel{\text{def}}{=} \bigcup_{x \in H} x \downarrow, \quad x \downarrow \stackrel{\text{def}}{=} \{ y \in D_\surd : y \sqsubseteq x \}.$$

**Lemma 4.16** *The operators $\hat{\omega} : \mathcal{P}_\surd(D)^k \to \mathcal{P}_\surd(D)$ are continuous.*

**Proof**: easy verification. Note that we deal with the leftclosure of the sets $\omega_\surd(\tilde{x})$. □

Using the semantic operators $\mathbin{\hat{;}}$, $\hat{\omega}$ and the union for modelling nondeterminism we get a least compositional meaning function

$$Me^{\text{cpo}} : Prog^\sigma(\Omega) \to \mathcal{P}_\surd(D).$$

Here the action symbols $\alpha \in Act$ are interpreted by the sets $\hat{\alpha} \stackrel{\text{def}}{=} \{ \alpha_D\surd \} \downarrow$.

**Example 4.17** The trace semantics of the program $P = \xi + \beta$ where $\sigma(\xi) = \alpha;\xi$ is the set $\{\alpha^n : n \geq 1\} \cup \{\bot, \beta, \beta\surd\}$. The trace semantics of $Q = \gamma_1 \parallel \gamma_2$ is the set

$$\{ \bot,\ \gamma_1,\ \gamma_2,\ \gamma_1\gamma_2,\ \gamma_2\gamma_1,\ \gamma_1\gamma_2\surd,\ \gamma_2\gamma_1\surd \}.$$

Hence we get the trace semantics of $P;Q$ by applying the operator $\hat{;}$:

$$\{\alpha^n : n \geq 1\} \cup \{ \bot,\ \beta,\ \beta\gamma_1,\ \beta\gamma_2,\ \beta\gamma_1\gamma_2,\ \beta\gamma_2\gamma_1,\ \beta\gamma_1\gamma_2\surd,\ \beta\gamma_2\gamma_1\surd \}$$

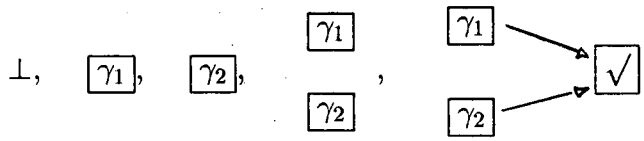
The pomset semantics of $P$ consists of the pomsets

$$p_n \stackrel{\text{def}}{=} \underbrace{\boxed{\alpha} \longrightarrow \boxed{\alpha} \longrightarrow \ldots \longrightarrow \boxed{\alpha}}_{n}$$

and the pomsets

$$\bot,\quad \boxed{\beta},\quad \boxed{\beta} \longrightarrow \boxed{\surd}.$$

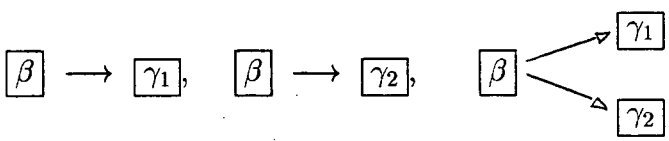
The pomset semantics of $Q$ consists of the five pomsets



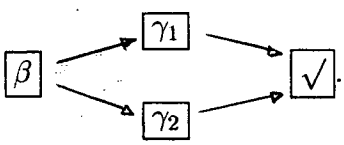

Hence the pomset semantics of $P;Q$ is the set which consists of the pomsets $p_n$, $n \geq 1$, the pomsets

$$\bot,\quad \boxed{\beta},\quad \boxed{\beta} \longrightarrow \boxed{\surd}$$

and the pomsets




and




□

# 5 The connection between the metric and partial order approach

In this section we show that if a semantic domain $A$ for the sublanguage $Prog_{\text{fin}}$ is given such that both, the metric and the partial order approach, can be applied then the

partial order semantics is an abstraction of the metric semantics. We make the following assumptions about $A$:

$A$ is a set that is endowed with a partial order $\sqsubseteq$ such that $D = (A, \sqsubseteq)$ is a pointed poset as in section 4. We assume that there is a metric on $M = A \setminus \{\bot\}$ is induced by a ranking $x \mapsto x[n]$, $n \geq 1$, such that $x[n]$ is the greatest element of the set

$$\downarrow^n (x) \stackrel{\text{def}}{=} \{y \in D : y \sqsubseteq x,\ \rho(y) \leq n\}.$$

We put $x[0] = \bot$ and $\bot[n] = \bot$, $\rho(\bot) = 0$. We require that for all $x \in \overline{M}$:

$$\rho(x[n]) = \min\{\rho(x),\ n\}$$

In addition we assume that there are semantic operators $;_A$ and $\omega_A$ on $A$ and interpretations $\alpha_A \in A$ of the atomic actions such that

- $\alpha_A \in M$
- Whenever $x \in M$, $y \in A$ then $x ;_A y \in M$.
- Whenever $x_1, \ldots, x_k \in M$ then $\omega_A(x_1, \ldots, x_k) \subseteq M$.

We require that $;_A$ and $\omega_A$ as operators on $M$ satisfies the conditions given in section 3 (see Definition 3.3 and 3.4) and that $;_A$ as an operator on the partial order $D$ satisfies the conditions given in Definition 4.2. We make some additional requirements about the semantic operators $\omega_A$. We require that for each $k$-ary operator symbol $\omega \in \Omega$ and for all $x_1, \ldots, x_k, y_1, \ldots, y_k \in A^k$:

(1) If $z \in \omega_A(x_1, \ldots, x_k)$ then $\rho(z) \geq \max\{\rho(x_i) : i = 1, \ldots, k\}$.

(2) $\omega_A$ is monotone w.r.t. the lower preorder $\sqsubseteq_L$ on $\mathcal{P}_{\text{fin}}(D)$:

$$x_i \sqsubseteq y_i,\ i = 1, \ldots, k \implies \omega_A(x_1, \ldots, x_k) \sqsubseteq_L \omega_A(y_1, \ldots, y_k)$$

where $H \sqsubseteq_L I$ iff $\forall x \in H\ \exists y \in I\ .\ x \sqsubseteq y$.

Then it can be shown that for all $x_1, \ldots, x_k \in \overline{M}$ and $z \in \omega_{\overline{M}}(x_1, \ldots, x_k)$:

$$\rho(z) \geq \max\{\rho(x_i) : i = 1, \ldots, k\}$$

Then for all $x, y \in A$:

$$\bot = x[0] \sqsubseteq x[1] \sqsubseteq \ldots \sqsubseteq x[n] \sqsubseteq \ldots \sqsubseteq x$$

$$x \sqsubseteq y \implies x[n] \sqsubseteq y[n]$$

**Remark 5.1** Suitable semantic operators for modelling parallelism without communication, hiding, relabeling or prefixing would fulfill the conditions (1) and (2). E.g. dealing with $A = Act^*$ or $A = Pom^*$ the parallel operator $\|$ satisfies both conditions. Problems arises when operators like $CCS$-restriction or parallelism with $TCSP$-communication

should be modelled. For instance the $CCS$-restriction operator $P \mapsto P \setminus L$ which forbids the execution of actions $\alpha \in L$ has a natural interpretation in $A = Act^*$:

$$Act^* \to Act^*, \quad x \setminus L = \alpha_1 \ldots \alpha_m$$

where $x = \alpha_1 \ldots \alpha_n$ and $m = \max \{ N : \alpha_1, \ldots, \alpha_N \notin L \}$ and $\max \emptyset = 0$. This operator (identified with the operator $Act^* \to \mathcal{P}_{\text{fin}}(Act^*)$, $x \mapsto \{x \setminus L\}$) violates the first condition. E.g. for $x = \alpha\beta$ and $L = \{\beta\}$ we have $x \setminus L = \alpha$. Then

$$\rho(x \setminus L) = 1 < 2 = \rho(x).$$

The $TCSP$-like communication $\|_L$ also yields problems: $P \|_L Q$ describes a process that executes $P$ and $Q$ in parallel where $P$ and $Q$ are enforced to communicate on all actions $\alpha \in L$. The natural interpretation of the program $\alpha;\gamma \|_{\{\alpha\}} \beta$ in $Act^*$ would be the single-element set $\{\beta\}$. This is because the process $\alpha;\gamma$ has no chance to execute its first action $\alpha$. Then condition (1) is violated since

$$\rho(\beta) = 1 < 2 = \max \{ \rho(\alpha\gamma), \rho(\beta) \}.$$

□

We define a function $\varphi : \mathcal{P}_{\text{co}}(\overline{M}) \to \mathcal{P}_{\surd}(D)$ such that $\varphi(Me^{\text{cms}}(P)) = Me^{\text{cpo}}(P)$. The function $\varphi$ replaces each infinite computation by its finite subcomputations and each terminating computation $x$ by $x\surd$ and all its predecessors.

**Definition 5.2** *Let* $\varphi : \mathcal{P}_{\text{co}}(\overline{M}) \to \mathcal{P}_{\surd}(D)$ *be given by:*

$$\varphi(H) \stackrel{\text{def}}{=} \downarrow^{\text{fin}}(H) \cup \{x\surd : x \in H \cap A\}$$

*where*

$$\downarrow^{\text{fin}}(x) \stackrel{\text{def}}{=} \bigcup_{n \geq 0} x[n]\downarrow, \quad \downarrow^{\text{fin}}(H) = \bigcup_{x \in H} \downarrow^{\text{fin}}(x)$$

*for each* $x \in \overline{M}$ *and each subset* $H$ *of* $\overline{M}$.

**Example 5.3** Let $H \in \mathcal{P}_{\text{co}}(Act^\infty)$ be the set consisting of the infinite trace $\alpha\alpha\alpha\ldots$ and the trace $\beta$. Then the infinite trace $\alpha\alpha\alpha\ldots$ is substituted by its finite prefixes $\alpha^n$. The terminating computation desrcibed by the finite trace $\beta$ is replaced by the prefixes of $\beta\surd$. Hence:

$$\varphi(H) = \{ \alpha^n : n \geq 1 \} \cup \{ \bot, \beta, \beta\surd \}$$

□

We now present the main result of this section: The denotational linear time semantics defined in the partial order approach is an abstraction of the denotational linear time semantics defined in the metric approach. Given the metric semantics $Me^{\text{cms}}(P)$ one gets the partial order semantics $Me^{\text{cpo}}(P)$ by substituting infinite behaviour by its finite subcomputations and by substituting finite behaviour $x \in M$ by the subcomputations of $x\surd$.

It should be noted that in general a reverse mapping $\Psi$ with $\Psi \circ Me^{\text{cpo}} = Me^{\text{cms}}$ does not exist. In the case of pomsets e.g. this can be seen by considering the programs $P = \alpha \parallel \xi$ and $P' = P + \xi$ with $\sigma(\xi) = \alpha;\xi$. The pomsets associated with $P$ and $P'$ in the cpo setting coincide but are different in the metric setting.

**Theorem 5.4** $\varphi \circ Me^{\text{cms}} = Me^{\text{cpo}}$

**Proof**: As shown in [1]: $Me^{\text{cms}}$ is the unique fixed point of the contracting mapping

$$\Psi : (Prog \to \mathcal{P}_{\text{co}}(\overline{M})) \to (Prog \to \mathcal{P}_{\text{co}}(\overline{M}))$$

which is given by:

- $\Psi(f)(\alpha) = \overline{\alpha}$
- $\Psi(f)(\xi) = f(\sigma(\xi))$
- $\Psi(f)(P_1 + P_2) = \Psi(f)(P_1) \cup \Psi(f)(P_2)$
- $\Psi(f)(P_1; P_2) = \Psi(f)(P_1) \,\bar{;}\, \Psi(f)(P_2)$
- $\Psi(f)(\omega(P_1, \ldots, P_k)) = \overline{\omega}(\, \Psi(f)(P_1), \ldots, \Psi(f)(P_k))\,)$

By Banach's fixed point theorem $Me^{\text{cms}} = \lim f_n$ where $f_1 : Prog \to \mathcal{P}_{\text{co}}(\overline{M})$ is an arbritary function and $f_{n+1} = \Psi(f_n)$.

$Me^{\text{cpo}}$ is the least fixed point of the continuous mapping

$$\Phi : (Prog \to \mathcal{P}_{\surd}(D)) \to (Prog \to \mathcal{P}_{\surd}(D))$$

which is given by:

- $\Phi(g)(\alpha) = \widehat{\alpha}$
- $\Phi(g)(\xi) = g(\sigma(\xi))$
- $\Phi(g)(P_1 + P_2) = \Phi(g)(P_1) \cup \Phi(g)(P_2)$
- $\Phi(g)(P_1; P_2) = \Phi(g)(P_1) \,\hat{;}\, \Phi(g)(P_2)$
- $\Phi(g)(\omega(P_1, \ldots, P_k)) = \widehat{\omega}(\, \Phi(g)(P_1), \ldots, \Phi(g)(P_k))\,)$

By Tarski's fixed point theorem $Me^{\text{cpo}} = \bigsqcup g_n$ where $g_0 : Prog \to \mathcal{P}_{\surd}(D)$ is given by $g_0(P) = \{\bot\}$ for all $P \in Prog$ and $g_{n+1} = \Phi(g_n)$. In the sequel we fix this definition of the functions $g_n$.

Before going into detail we give a sketch of the proof for Theorem 5.4: We show that there is a function $f_1 : Prog \to \mathcal{P}_{\text{co}}(\overline{M})$ with $\varphi \circ f_1 = g_2$. Then we conclude that $\varphi \circ f_n = g_{n+1}$ for all $n \geq 1$. Since $\varphi$ maps the limit of a Cauchy sequence $(H_n)$ in $\mathcal{P}_{\text{co}}(\overline{M})$ whose image is monotone to the least upper bound of $(\varphi(H_n))$ in $\mathcal{P}_{\surd}(D)$ (Claim 1) we get for all $P \in Prog$:

$$\varphi(\, Me^{\text{cms}}(P)\,) = \varphi\Big(\lim_{n\to\infty} f_n(P)\Big)$$

$$= \bigsqcup_{n\geq 1} \varphi(f_n(P)) = \bigsqcup_{n\geq 1} g_{n+1}(P) = Me^{\text{cpo}}(P)$$

**Claim 1** *If $(H_n)_{n\geq 1}$ is a Cauchy sequence in $\mathcal{P}_{\mathrm{co}}(\overline{M})$ such that $\varphi(H_1) \subseteq \varphi(H_2) \subseteq \ldots$ then*

$$\varphi\Big(\lim_{n\to\infty} H_n\Big) = \bigcup_{n\geq 1} \varphi(H_n).$$

**Proof**: Let $H = \lim H_n$. We may assume w.l.o.g. that $d(H,H_n) \leq 1/2^n$ for all $n \geq 1$. Then $H[n] = H_n[n]$ for all $n \geq 1$. Let $I = \bigcup \varphi(H_n)$.

First we show $\varphi(H) \subseteq I$. Let $y \in \varphi(H)$.

Case 1: $y \in \downarrow^{\mathrm{fin}}(H)$. Then $y \sqsubseteq x[n]$ for some $x \in H$ and $n \geq 1$. There exists $x' \in H[n]$ with $x[n] = x'[n]$. Then $y \sqsubseteq x[n] = x'[n]$. Hence $y \in \downarrow^{\mathrm{fin}}(H_n) \subseteq I$.

Case 2: $y = x\surd$ where $x \in H \cap A$. Let $n = \rho(x)$. Then $x = x[n]$. Hence

$$x = x[n+1] \in H[n+1] = H_{n+1}[n+1]$$

Let $z \in H_{n+1}$ such that $x = z[n+1]$. Then

$$\min\{\ \rho(z),\ n+1\ \} = \rho(\ z[n+1]\ ) = \rho(x) = n$$

Hence $\rho(z) = n$ and $z = z[n] = (z[n+1])[n] = x[n] = x$. We conclude $x \in H_{n+1}$ and therefore

$$y = x\surd \in \varphi(H_{n+1}) \subseteq I.$$

Next we show $I \subseteq \varphi(H)$. Let $y \in I$.

Case 1: $y \in D$. Then $y \sqsubseteq x[m]$ for some $x \in H_n$ and $n, m \geq 1$. W.l.o.g. $m \geq n$. Since $\varphi(H_n) \subseteq \varphi(H_m)$ we have $x[m] \in \varphi(H_m)$. Hence $x[m] \sqsubseteq z[k]$ for some $z \in H_m$ and $k \geq 1$. Then $x[m] \sqsubseteq z[m]$. Since $H[m] = H_m[m]$ there exists $w \in H$ with $z[m] = w[m]$. Then $y \sqsubseteq w[m]$. Therefore $y \in \varphi(H)$.

Case 2: $y \notin D$. Then $y = x\surd$ where $x \in H_n \cap A$ for some $n \geq 1$. Let

$$m = \max\{\ \rho(x),\ n\ \} + 1.$$

Then $\varphi(H_n) \subseteq \varphi(H_m)$ and hence $x \in H_m$. Then $x = x[m] \in H_m[m] = H[m]$. Hence $x = z[m]$ for some $z \in H$. Then

$$\min\{\ \rho(z),\ m\ \} = \rho(z[m]) = \rho(x) < m$$

Hence $\rho(z) < m$ and therefore $z = z[m] = x \in H$. We get $y = x\surd \in \varphi(H)$. □

We now construct a function $f_1 : Prog \to \mathcal{P}_{\mathrm{co}}(\overline{M})$ such that $\varphi \circ f_1 = g_2$. The problem is that the definition of the sequence $(g_n)$ uses the bottom element of $\mathcal{P}_{\surd}(D)$ which does not belong to $M$. In order to give an adequate description of $\{\bot\}$ in the metric we extend $\mathcal{P}_{\mathrm{co}}(\overline{M})$ by the emptyset: Let $\mathcal{P}^{\emptyset}_{\mathrm{co}}(\overline{M})$ denote the collection of compact subsets of $\overline{M}$ including the emptyset. We extend the operators $\bar{;}$ and $\overline{\omega}$ on the emptyset:

$$\emptyset \bar{;} H \stackrel{\mathrm{def}}{=} \emptyset, \quad H \bar{;} \emptyset \stackrel{\mathrm{def}}{=} H, \quad \overline{\omega}(H_1,\ldots,H_k) \stackrel{\mathrm{def}}{=} \emptyset$$

if $H_i = \emptyset$ for some $i$. We also extend $\varphi$ to a function $\mathcal{P}^{\emptyset}_{\mathrm{co}}(\overline{M}) \to \mathcal{P}_{\surd}(D)$:

$$\varphi(\emptyset) \stackrel{\mathrm{def}}{=} \{\bot\}$$

The interpretation of the atomic actions $\alpha \in Act$ in $\mathcal{P}^{\emptyset}_{\mathrm{co}}(\overline{M})$ is the interpretation of $\alpha$ in $\mathcal{P}_{\mathrm{co}}(\overline{M})$. Using these semantic operators on $\mathcal{P}^{\emptyset}_{\mathrm{co}}(\overline{M})$ we get an extension

$$( \ Prog \ \rightarrow \ \mathcal{P}^{\emptyset}_{\mathrm{co}}(\overline{M}) \ ) \ \rightarrow \ ( \ Prog \ \rightarrow \ \mathcal{P}^{\emptyset}_{\mathrm{co}}(\overline{M}) \ )$$

of the operator $\Psi$ which we also denote by $\Psi$.

**Claim 2** *The function* $\varphi : \mathcal{P}^{\emptyset}_{\mathrm{co}}(\overline{M}) \rightarrow \mathcal{P}_{\surd}(D)$ *satisfies:*

- $\varphi(\overline{\alpha}) \ = \ \widehat{\alpha}$
- $\varphi(H_1 \cup H_2) \ = \ \varphi(H_1) \ \cup \ \varphi(H_2)$
- $\varphi(H_1 \ \overline{;} \ H_2) \ = \ \varphi(H_1) \ \widehat{;} \ \varphi(H_2)$
- $\varphi( \ \overline{\omega}(H_1, \ldots, H_k) \ ) \ = \ \widehat{\omega}( \ \varphi(H_1), \ \ldots, \ \varphi(H_k) \ )$

**Proof**: It is clear that $\varphi(\overline{\alpha}) \ = \ \widehat{\alpha}$ and that $\varphi$ is compositional w.r.t. the union.

Claim: $\varphi(H_1 \ \overline{;} \ H_2) \ \subseteq \ \varphi(H_1) \ \widehat{;} \ \varphi(H_2)$

Proof: Let $z \in \varphi(H_1 \ \overline{;} \ H_2)$. If $z \in A$ then $z \ \sqsubseteq \ x[n]$ for some $x \in H_1 \ \overline{;} \ H_2$ and $n \geq 1$. Hence $x \ = \ x_1 \ ;_{\overline{M}} \ x_2$ for some $x_i \in H_i$.

- If $\rho(x_1) \geq n$ then $x[n] \ = \ x_1[n]$. Hence $z \ \sqsubseteq \ x_1[n] \ ;_{\surd} \ \perp \ \in \ \varphi(H_1) \ \widehat{;} \ \varphi(H_2)$.
- If $\rho(x_1) \ = \ m \ < \ n$ then $x_1 \in \varphi(H_1)$ and

  $$z \ \sqsubseteq \ x[n] \ = \ x_1 \ ;_A \ x_2[n-m].$$

  Since $x_2[n-m] \in \varphi(H_2)$ we have: $z \in \varphi(H_1) \ \widehat{;} \ \varphi(H_2)$.

If $z \ = \ z'\surd$ then $z' \in \ A \ \cap \ (H_1 \ \overline{;} \ H_2)$. Hence there exist $x_1 \in \ A \cap H_1$ and $x_2 \in A \cap H_2$ such that $z' \ = \ x_1 \ ;_A \ x_2$. Then $x_1\surd \in \varphi(H_1)$ and $x_2\surd \in \varphi(H_2)$ and

$$z \ = \ z'\surd \ = \ x_1\surd \ ;_{\surd} \ x_2\surd \ \in \ \varphi(H_1) \ \widehat{;} \ \varphi(H_2)$$

Claim: $\varphi(H_1 \ \overline{;} \ H_2) \ \supseteq \ \varphi(H_1) \ \widehat{;} \ \varphi(H_2)$

Proof: Let $z \in \varphi(H_1) \ \widehat{;} \ \varphi(H_2)$. If $z \in A$ then $z \ = \ x_1 \ ;_{\surd} \ x_2$ for some $x_1 \in \varphi(H_1)$ and some $x_2 \in \varphi(x_2)$.

- If $x_1 \ \in \ D$ then $x_1 \ \sqsubseteq \ x_1'[n]$ for some $x_1' \in H_1$ and $n \geq 1$. Then $z \ = \ x_1$ and

  $$z \ = \ x_1 \ \sqsubseteq \ (x_1' \ ;_{\overline{M}} \ x_2')[n]$$

  where $x_2'$ is an arbritary element of $H_2$. Hence

  $$z \in \ \downarrow^{\mathrm{fin}}( \ H_1 \ \overline{;} \ H_2 \ ) \ \subseteq \ \varphi(H_1 \ \overline{;} \ H_2).$$

- If $x_1 = x_1'\surd$ where $x_1' \in H_1 \cap A$ then $x_2 \in A$ (otherwise $z \in D_\surd \setminus A$). Hence $x_2 \sqsubseteq x_2'[n]$ for some $x_2' \in H_2$ and $n \geq 1$. Then

$$z = x_1' ;_A x_2 \sqsubseteq x_1' ;_A x_2'[n]$$

Let $m = \rho(z)$ and w.l.o.g. $n \geq m$. If $\rho(x_1') \geq m$ then

$$z = z[m] = x_1'[m] = (x_1' ;_{\overline{M}} x_2')[m] \in \varphi(H_1 \bar{;} H_2).$$

Otherwise $\rho(x_1') = k < m$ and

$$z = z[m] = x_1' ;_A x_2[m-k] \sqsubseteq (x_1' ;_A x_2'[n])[m]$$

$$= (x_1' ;_{\overline{M}} x_2')[m] \in \varphi(H_1 \bar{;} H_2).$$

Since $\varphi(H_1 \bar{;} H_2)$ is leftclosed we get $z \in \varphi(H_1 \bar{;} H_2)$.

Claim: $\varphi(\,\overline{\omega}(H_1, \ldots, H_k)\,) = \widehat{\omega}(\,\varphi(H_1), \ldots, \varphi(H_k)\,)$

Proof: For simplicity we assume $k = 1$.

First we show '$\subseteq$': Let $z \in \varphi(\overline{\omega}(H))$. If $z \in A$ then $z \sqsubseteq x[n]$ where $x \in \omega_{\overline{M}}(y)$ for some $y \in H$ and some $n \geq 1$. W.l.o.g. $n \geq \rho(z)$. Since

$$\omega_{\overline{M}}(y)[n]\downarrow = \omega_A(\,y[n]\,)[n]\downarrow \subseteq \omega_A(\,y[n]\,)\downarrow \subseteq \omega_\surd(\,y[n]\,)\downarrow$$

and since $y[n] \in \varphi(H)$ we get:

$$z \in \omega_\surd(y[n])\downarrow \subseteq \widehat{\omega}(\varphi(H)).$$

If $z = z'\surd$ where $z' \in \overline{\omega}(H) \cap A$ then there is some $y \in H$ and a sequence $(z_n)$ in $A$ with $z_n = z_{n+1}[n]$ and

$$z_n \in \omega_A(y[n])[n] = \omega_{\overline{M}}(y)[n]$$

such that $z' = \lim z_n$. Let $n = \rho(z') + 1$. Then $z' = z_n = z_{n+1} = \ldots$ and

$$z' \in \omega_{\overline{M}}(y)[n].$$

Hence $z' = z''[n]$ for some $z'' \in \omega_{\overline{M}}(y)$. Since

$$\min\{\,\rho(z''),\, n\,\} = \rho(z''[n]) = \rho(z') < n$$

we have $\rho(z'') < n$. Therefore $z' = z''[n] = z'' \in \omega_{\overline{M}}(y)$. Since $\rho(z') \geq \rho(y)$ we have $y \in H \cap A$. Hence $y\surd \in \varphi(H)$ and

$$z = z'\surd \in \omega_\surd(y\surd).$$

Therefore $z \in \widehat{\omega}(\varphi(H))$.

We show '$\supseteq$': Let $z \in \widehat{\omega}(\varphi(H))$. If $z \in A$ then $z \in \omega_A(y)\downarrow$ for some $y \in \varphi(H) \cap A$. Then $z \sqsubseteq x$ for some $x \in \omega_A(y)$. Since $y \in \varphi(H) \cap A$ there is some $y' \in H$, $N \geq 1$ with $y \sqsubseteq y'[N]$. Then for all $n \geq N$:

$$y \sqsubseteq y'[N] \sqsubseteq y'[n]$$

and therefore

$$\omega_A(y) \sqsubseteq_L \omega_A(\, y'[N]\,) \sqsubseteq_L \omega_A(\, y[n]\,).$$

Hence there exists a sequence $(x_n)_{n\geq N}$ with $x_n \in \omega_A(y'[n])$ and $x \sqsubseteq x_n$ for all $n \geq N$. Since

$$x_n[n] \in \omega_A(y'[n])[n] = \omega_{\overline{M}}(y')[n] \subseteq \overline{\omega}(H)[n]$$

there is a sequence $(x'_n)_{n\geq N}$ in $\overline{\omega}(H)$ with $x_n[n] = x'_n[n]$. Since $\overline{\omega}(H)$ is compact there is a convergent subsequence $(x'_{n_m})_{m\geq N}$. We put:

$$x' \stackrel{\text{def}}{=} \lim_{m\to\infty} x'_{n_m}$$

Then $x' \in \overline{\omega}(H)$. Let $m = \rho(x)$. Then $x'[m] = x'_{n_l}[m]$ for some $l \geq N$. Hence

$$x'[m] = x_{n_l}[m]$$

for all $l \geq N$. Since $x \sqsubseteq x_{n_l}$ and $\rho(x) = m$ we have:

$$z \sqsubseteq x \sqsubseteq x_{n_l}[m] = x'[m].$$

Therefore $z \in \varphi(\overline{\omega}(H))$.

If $z = z'\surd$ where $z' \in A$ then $z' \in \omega_A(y')$ for some $y' \in H \cap A$. Then

$$z' \in \omega_A(y') = \omega_{\overline{M}}(y') \subseteq \overline{\omega}(H)$$

Hence $z = z'\surd \in \varphi(\overline{\omega}(H))$. □

If $X$ is a set endowed with semantic operators $;'$, $+' : X \times X \to X$ and $\omega' : X^k \to X$ (where $k$ is the arity of $\omega \in \Omega$) and interpretations $\alpha' \in X$ of the atomic actions then by a homomorphism on $X$ we mean a function $f : Prog \to X$ such that:

- $f(\alpha) = \alpha'$
- $f(P_1 + P_2) = f(P_1) +' f(P_2)$
- $f(P_1; P_2) = f(P_1) ;' f(P_2)$
- $f(\omega(P_1, \ldots, P_k)) = \omega'(\, f(P_1), \ldots, f(P_k)\,)$

Given a function $F : Idf \to X$ there is a unique homomorphism $f : Prog \to X$ with $f(\xi) = F(\xi)$ for all $\xi \in Idf$. Dealing with $X = \mathcal{P}_{\surd}(D)$ we have for all $n \geq 1$: $g_n : Prog \to \mathcal{P}_{\surd}(D)$ is the unique homomorphism such that $g_n(\xi) = g_{n-1}(\sigma(\xi))$.

**Claim 3** *Let $f : Prog \to \mathcal{P}^{\emptyset}_{\text{co}}(\overline{M})$ and $g : Prog \to \mathcal{P}_{\surd}(D)$ be homomorphisms such that $\varphi \circ f = g$. Then $\varphi \circ \Psi(f) = \Phi(g)$.*

**Proof**: By structural induction it can be shown that $\varphi(\, \Psi(f)(P)\,) = \Phi(g)(P)$. Uses Claim 2. □

**Claim 4** *If $f : Prog \to \mathcal{P}^{\emptyset}_{\text{co}}(\overline{M})$ is a homomorphism on $\mathcal{P}^{\emptyset}_{\text{co}}(\overline{M})$ then $f(G) \in \mathcal{P}_{\text{co}}(\overline{M})$ for each guarded statement $G$.*

**Proof** easy verification. Uses structural induction and $\overline{\alpha} \in \mathcal{P}_{\mathrm{co}}(\overline{M})$, $H \bar{;} I \neq \emptyset$ if $H \neq \emptyset$. $\square$

Let $f_0 : Prog \to \mathcal{P}^{\emptyset}_{\mathrm{co}}(\overline{M})$ be the unique homomorphism with $f_0(\xi) = \emptyset$ for all $\xi \in Idf$. Using Claim 2 and the fact that $g_1$ is the unique homomorphism $Prog \to \mathcal{P}_{\surd}(D)$ with $g_1(\xi) = \{\bot\} = \varphi(f_0(\xi))$ we get $\varphi \circ f_0 = g_1$. Let $F : Prog \to \mathcal{P}^{\emptyset}_{\mathrm{co}}(\overline{M})$ be the unique homomorphism such that $F(\xi) = f_0(\sigma(\xi))$ for all $\xi \in Idf$. Then we get by structural induction and Claim 4:

$$F(P) \in \mathcal{P}_{\mathrm{co}}(\overline{M}) \text{ for all } P \in Prog.$$

Since $F = \Psi(f_0)$ we get by Claim 3: $\varphi \circ F = g_2$. Let $f_1 : Prog \to \mathcal{P}_{\mathrm{co}}(\overline{M})$ be given by $f_1(P) = F(P)$ for all $P \in Prog$. Then $\varphi \circ f_1 = g_2$. Let $f_n = \Psi(f_{n-1})$ for all $n \geq 2$. Then by Claim 3: $\varphi \circ f_n = g_{n+1}$ for all $n \geq 1$. $\square$

On might suppose that the proof of Theorem 5.4 is too complicate and it would be easier if $\bot$ is not excluded from $M$. The problem is that then because of the natural assumption $\bot ;_A x = x ;_A \bot = x$ the resulting sequence operator $\bar{;}$ on $\mathcal{P}_{\mathrm{co}}(\overline{M})$ would not be contracting in its second argument. This is essential for the definition of the semantics $Me^{\mathrm{cms}}$.

**Example 5.5** Theorem 5.4 yields the consistency of the trace semantics on $\mathcal{P}_{\downarrow}(Act^*_{\surd})$ of [11] and the trace semantics on $\mathcal{P}_{\mathrm{co}}(Act^{\infty})$ of [5]. We also obtain the consistency of the pomset semantics on $\mathcal{P}_{\surd}(Pom^*)$ of [7] and the pomset semantics on $\mathcal{P}_{\mathrm{co}}(Pom^{\infty})$ of [4]. $\square$

# 6 Conclusion

We presented a general framework to define denotational linear time semantics for languages that allow for nondeterminism, recursion and sequential composition. We gave conditions that a 'good' sequence operator on a domain $A$ for finite behaviour has to fulfill (Definition 3.3 and Definition 4.2). In these characterizations of a good sequence operator it is essential that we do not deal with deadlocked processes. In order to deal with the case where processes may deadlock the semantic domain $A$ has to be devided into elements representing successfull terminating computations and elements for representing deadlocked computations. In the partial order approach one has to require that the set of successfull terminating computations is leftclosed (which asserts that no partial execution of a successfull terminating process can be deadlocked). A good sequence operator on $A$ is then an operator $;_A$ which satisfies:

- $x ;_A y = x$ for each deadlocked computations $x$
- $x ;_A y$ represents a successfull terminating computation if and only if $x$ and $y$ represents successfull terminating computations
- In the metric case: If $x$ stands for a successfull terminating computation then the $n$-cuts $x[n]$ of $x$ also stand for successfull terminating computations and

$$(x ;_A y)[n] = \begin{cases} x[n] & : \text{ if } \rho(x) \geq n \\ x ;_A y[n-m] & : \text{ if } \rho(x) = m < n. \end{cases}$$

- In the partial order case $;_A$ has to fulfill the conditions of Definition 4.2.

Then in the metric case one can define a denotational linear time semantics on $\mathcal{P}_{\mathrm{co}}(\overline{M})$ as in section 3: take the canonical extension $;_{\overline{M}}$ of $;_A$ on the completion $\overline{M}$ of $A$ as semantic operator for modelling sequential composition on finite and infinite computations and use the operator

$$H \,\bar{;}\, I \stackrel{\text{def}}{=} \{\, x ;_{\overline{M}} y \,:\, x \in H,\, y \in I \,\}$$

on $\mathcal{P}_{\mathrm{co}}(\overline{M})$. In the partial order approach a little modification of the semantic domain $\mathcal{P}_{\surd}(D)$ is needed. Define $D_{\surd}$ to be the set of all elements $x \in D$ and new elements $x\surd$ where $x \in D$ stands for a successfull terminating computation. Then a suitable sequence operator on $\mathcal{P}_{\surd}(D)$ can be defined as in section 4.

The assumptions in section 5 are closely related to the requirement that the underlying partial order is endowed with a finite weight $\rho$ in the sense of [2]. A finite weight means a function $\rho : D \to I\!N_0$ such that:

(1) $\rho(\bot) = 0$

(2) If $x \sqsubseteq y$ then $\rho(x) \leq \rho(y)$.

(3) For each $x \in D$ the set $\{\, y \in D \,:\, y \sqsubseteq x,\ \rho(y) \leq n \,\}$ has a greatest element (denoted by $x[n]$).

Then $x \mapsto x[n]$ is a ranking on $M = D \setminus \{\bot\}$ as in section 3. In section 5 we do not need condition (2). Nevertheless (2) is a natural requirement since we interpret $x \sqsubseteq y$ as $x$ is a subcomputation of $y$. Hence $\rho(x)$, the length of the execution of $x$, is at most $\rho(y)$. Instead of (2) we require

$$\rho(\, x[n] \,) = \min \{\, \rho(x),\ n \,\}.$$

The treatment of sequential composition in branching time semantics is more complicate. A suitable sequence operator $;_A$ on a branching time model $A$ has to be defined in such a way that $x ;_A y$ arises from $x$ by 'appending' $y$ at every 'maximal computation' of $x$ (cf. e.g. the sequence operator on prime event structures [1]). We do not see a way to formalize the 'maximal computations' of the elements in branching time models. Hence, we cannot propose a general framework for the treatment of sequential composition in branching time models.

# References


[1] C. Baier, M.E. Majster-Cederbaum: Denotational semantics in the cpo and metric approach, Theoretical Computer Science, Vol. 135, pp 171-220, 1995.

[2] C. Baier, M.E. Majster-Cederbaum: Construction of a cms on a given cpo, submitted for publication, Techn. Report 28/95, Reihe Informatik, Universität Mannheim, 1995.

[3] J.W. de Bakker, J.I.Zucker: Processes and the Denotational Semantics of Concurrency, Information and Control, Vol.54, No. 1/2, pp 70-120, 1982.

[4] J.W. de Bakker, J.H.A. Warmerdam: Metric pomset semantics for a concurrent language with recursion, Report CS-R9033, Centre for Mathematics and Computer Science, Amsterdam, July 1990.

[5] J.W. de Bakker, J. Meyer: Metric semantics for concurrency, Report CS-R8803, Centre for Mathematics and Computer Science, Amsterdam, 1988.

[6] F. van Breugel: Topological Models in Comparative Semantics, Ph.D.Thesis, Vrije Universiteit Amsterdam, 1994.

[7] M. Broy: Operational and Denotational Semantics with Explicit Concurrency, Fundamenta Informaticae, Vol. 16, 1992.

[8] K. Bruce, J.C. Mitchell: PER Models of Subtyping, Recursive Terms and Higher-order Polymorphism, Journal of ACM, 8/92, 1992.

[9] H. Ehrig, F. Parisi-Presicce, P. Boehm, C. Rieckhoff, C. Dimitrovici, M. Große-Rohde: Combining Data Type Specifications using Projection Algebras, Theoretical Computer Science, Vol. 71, 1990.

[10] H. Hahn: Reelle Funktionen, Chelsea, New York, 1948.

[11] C.A.R. Hoare: Communicating Sequential Processes, Prentice Hall, 1985.

[12] K. Kuratowski: Sur une méthode de métrisation complète des certains espaces d'ensembles compacts, Fundamentae Mathematicae, Vol. 43, pp 114-138, 1956.

[13] R. Milner: Communication and Concurrency, Prentice Hall, 1989.

[14] G.D. Plotkin: A Powerdomain Construction, SIAM Journal of Computation, Vol. 5, No. 3, pp 452-487, 1976.

[15] V. Pratt: The Pomset Model of Parallel Processes: Unifying the Temporal and the Spatial, Seminar on Concurrency, Lecture Notes in Computer Science 197, Springer-Verlag, 1984.

[16] M.B. Smyth: Power Domains, Journal of Computer and System Sience, Vol. 16, pp 23-36, 1978.