Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

J.N. Kok, J.J.M.M. Rutten

Contractions in comparing concurrency semantics

Computer Science/Department of Software Technology Report CS-R8755 November

The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

Contractions in Comparing Concurrency Semantics

Joost N. Kok, Jan J.M.M. Rutten
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

We define for a number of concurrent imperative languages both operational and denotational semantic models as fixed points of contractions on complete metric spaces. Next, we develop a general method for comparing different semantic models by relating their defining contractions and exploiting the fact that contractions have a unique fixed point.

1980 Mathematical Subject Classification: 68B10, 68C01.
1986 Computing Reviews Categories: D.3.1, F.3.2, F.3.3.
Key words and phrases: concurrency, imperative languages, denotational semantics, operational semantics, metric spaces, contractions, semantic equivalence.
Note: This work was carried out in the context of ESPRIT project 415: Parallel Architectures and Languages for Advanced Information Processing — a VLSI-directed approach.

0. Introduction

We present a study of three concurrent imperative languages, called L_0, L_1, and L_2. For each of them we shall define an *operational semantics* $\mathcal{O}_i$ and a *denotational semantics* $\mathcal{D}_i$, for $i = 0,1,2$, and give a comparison of the two models. (We shall use the terms *semantics* and *semantic model* as synonyms.) This *comparison* is the main subject of our paper, rather than the specific nature of the languages themselves, or the particular properties of their semantics.

The languages L_i have been defined and studied already in much detail in [BMOZ1,2] and [BKMOZ]. We rely heavily on these papers, using many definitions taken from them literally, and others in an adapted version. (The languages L_0, L_1, and L_2 we use here are called L_0, L_2, and L_3 in the papers mentioned.)

Let us try to characterize in a few words the languages under consideration. They all belong to the wide class of *concurrent* (*parallel*) *imperative* programming languages. We shall discuss parallel execution through interleaving (*shuffle*) of elementary actions (in L_0), together with *synchronization* and *communication* (in L_1) and extended with (an elementary form of) *message passing* (in L_2). Imperative concurrency is further characterized by an explicit operator for parallel composition on top of the usual imperative constructs, such as elementary action and sequential composition. Herein it differs from another widely used style, so-called *applicative* concurrency, where the parallelism is implicit. Further, L_0 and L_1 are *uniform* and L_2 is *nonuniform*. In L_0 and L_1 the elementary actions are left atomic, whereas in L_2 an interpretation of these actions is supplied. They consist of assignments, test and send and receive actions. Another important feature is the presence of *local nondeterminacy* (in L_0) and *global nondeterminacy* (in L_1 and L_2). (Sometimes this is called *internal* and *external* nondeterminacy.) The difference between the two has major implications for the different semantic models. (For an extensive discussion of this matter see, e.g., the introduction of [BKMOZ].)

For our semantic definitions we shall use *metric* structures, rather than order-theoretic domains. The metric approach is particularly felicitous for problems where histories, computational traces and tree-

like structures of some kind are essential. Moreover, it allows for the definition of the notion of *contraction*, which we discuss in more detail in a moment. Our operational models $\mathcal{O}_i$ are based on the transition system technique of Hennessy and Plotkin [HP] and Plotkin [Pl2, Pl3]. They are closely related to the ones defined in [BKMOZ], but there are some differences. We use labeled transitions and (in $\mathcal{O}_1$ and $\mathcal{O}_2$) communication is treated somewhat differently. Our denotational models D_i are almost exactly the same as in [BKMOZ]. They are defined compositionally, giving the meaning of a compound statement in terms of the meaning of its components, and tackling recursion with the help of fixed points. For D_1 and D_2 we use a reflexive domain, being a solution of some domain equation in the style of Plotkin [Pl1] and Scott [Sc]. We shall not give the details of solving in a metric setting this type of equations, but refer the reader to [BZ], where a solution was presented first, and to [AR], where this metric approach is reformulated and extended in a category-theoretic setting.

Although the semantic models presented here are (roughly) the same as in [BKMOZ], there is one major difference, being the way in which they are defined. In this paper we define both the operational and denotational models as fixed points of contractions.

A *contraction* $f:M\to M$ on a complete metric space M has the useful property that there exists one and only one fixed point $x\in M$ (satisfying $f(x)=x$). This elementary fact is known as Banach's fixed point theorem (see A.4.(b)). Such a fixed point x is entirely determined by the definition of f: any other element $y\in M$ satisfying the same properties as x, that is, satisfying $f(y)=y$, is equal to x. The contractions Φ we use in this paper are always of type

$$\Phi:(M_1\to M_2)\to(M_1\to M_2),$$

that is, they are defined on a complete metric function space $M_1\to M_2$. Then the fixed point of Φ is a function from M_1 to M_2.

The fact that our denotational models can be obtained as fixed points of suitable contractions is not very surprising, fixed points playing traditionally an important role in denotational semantics. It is interesting, however, to observe that the same method applies to the definition of *operational* models. One might wonder whether the models thus obtained still deserve to be called operational. That this is the case follows from the fact that they equal the models defined in the usual manner, without the use of fixed points (see lemma 1.12).

The main advantage of this style of defining semantic models as fixed points is that it enables us to compare them more easily. This brings us to the discussion of what has been announced above to be the main subject of this paper: the *comparison* of operational and denotational semantic models, which we shall also call the study of their *semantic equivalence*. About the question why this would be an interesting problem we want to be brief. Different semantic models of a given language can be regarded as different views of the same object. So they are in some way related. Their precise relationship we want to capture in some formal statement.

Let us now sketch the way we use contractions in our study of semantic equivalences. Let L be a language. Suppose an operational model $\mathcal{O}$ for L is given as the fixed point of a contraction

$$\Phi:(L\to M)\to(L\to M),$$

where M is a complete metric space. Suppose furthermore that we have a denotational model $\mathcal{D}$ for L of the same type as $\mathcal{O}$, that is, with $\mathcal{D}:L\to M$, for which we can prove $\Phi(\mathcal{D})=\mathcal{D}$. Then it follows from the uniqueness of the fixed point of Φ that $\mathcal{O}=\mathcal{D}$.

In the context of *complete partial ordering structures* similar approaches exist (see, e.g., [HP] and [AP]). There, the operational semantics $\mathcal{O}$ can be characterized as the (with respect to the pointwise ordering)

smallest function $\mathcal{F}$ satisfying $\Phi(\mathcal{F})=\mathcal{F}$, for some continuous function Φ. Then it follows from $\Phi(\mathcal{D})=\mathcal{D}$ that $\mathcal{O}$ is smaller than $\mathcal{D}$. In order to establish $\mathcal{O}=\mathcal{D}$ it is proved that $\mathcal{O}$ satisfies the defining equations for $\mathcal{D}$, from which it follows that $\mathcal{D}$ is smaller than $\mathcal{O}$. Please note that within the metric setting we can omit the second part of the proof.

In general $\mathcal{O}$ and $\mathcal{D}$ have different types, that is, they are mappings from L to different mathematical domains. In the languages we consider, this difference is caused by the fact that recursion is treated in the denotational and operational semantics *with* and *without* the use of so-called environments, respectively. Therefore, $\mathcal{O}$ and $\mathcal{D}$ cannot be fixed points of the same contraction. Now suppose $\mathcal{O}$ and $\mathcal{D}$ are defined as fixed points of

$$\Phi:(L\to M_1)\to(L\to M_1), \quad \text{and} \quad \Psi:(L\to M_2)\to(L\to M_2)$$

respectively, where M_1 and M_2 are different complete metric spaces. Then we can relate $\mathcal{O}$ and $\mathcal{D}$ by defining an intermediate semantic model for L as the fixed point of a contraction

$$\Phi':(L\to M')\to(L\to M'),$$

and by relating Φ , Φ' and Ψ as follows. If we define

$$f_1:(L\to M_1)\to(L\to M'), \quad \text{and} \quad f_2:(L\to M_2)\to(L\to M'),$$

and we next succeed in proving the commutativity (indicated by *) of the next diagram:

$$\begin{array}{ccc} L\to M_1 & \overset{\Phi}{\to} & L\to M_1 \\ f_1\downarrow & *_1 & \downarrow f_1 \\ L\to M' & \overset{\Phi'}{\to} & L\to M' \\ f_2\uparrow & *_2 & \uparrow f_2 \\ L\to M_2 & \overset{\Psi}{\to} & L\to M_2 \end{array}$$

then we are able to deduce the following relation between $\mathcal{O}$ and $\mathcal{D}$:

$$f_2(\mathcal{D})=f_1(\mathcal{O}).$$

It is straightforward from $*_1$ and $*_2$, and the fact that Φ, Φ', and Ψ are contractions.

This will be the procedure we follow for the models $\mathcal{O}_0$ and $\mathcal{D}_0$ of L_0 in section 1. There f_1 and f_2 are such, that for closed statements (i.e., containing no free statement variables) $s\in L_0$, we have: $\mathcal{O}(s)=\mathcal{D}(s)$. Once this result has been achieved for L_0, it is straightforward to adapt the definitions, lemmas and theorems involved so as to deduce a similar result for L_1 and L_2. (For the latter languages there is one slight complication. It appears to be convenient to relate $L\to M_1$ and $L\to M_2$ via *two* intermediate types, $L\to M'$ and $L\to M''$.) In [BMOZ1,2] and [BKMOZ] there have already been given proofs for the semantic equivalence of operational and denotational models for L_0 and L_1. These proofs, however, are quite complicated and not so easy to understand. Furthermore, the proof for L_1 is much more complex than that for L_0, involving an intermediate ready-set domain.

The method of proving semantic equivalence as described above is general in the sense that it is applicable to very different languages, such as L_0, L_1 and L_2.

This paper has seven sections. You are now reading section 0, the introduction. It is followed by the treatment of L_0, L_1 and L_2 in sections 1, 2, and 3 respectively. Then, in section 4, some conclusions and remarks about future research are formulated. Section 5 gives the references and section 6, the appendix, gives the basic definitions of metric topology.

Acknowledgements
We are much indebted to Jaco de Bakker, John-Jules Meijer, Ernst-Rudiger Olderog, and Jeffrey Zucker, authors and co-authors of the papers [BMOZ1,2] and [BKMOZ], respectively, on which we have relied heavily. We are also grateful to Jaco de Bakker for his many comments and suggestions made during our work on this subject. We thank Pierre America for pointing out an error in the definition of guardedness (which caused us considerable trouble and therefore increased our insights). We acknowledge fruitful discussions on our work in the Amsterdam concurrency group, including Jaco de Bakker, Frank de Boer, Arie de Bruin, John-Jules Meijer, and Erik de Vink. Finally, we express our thanks to Dini Verloop, who has expertly typed this document.

1. A SIMPLE LANGUAGE (L_0)

1.1 *Syntax*

For the definition of the first language studied in this paper, we need two sets of basic elements. Let A, with typical elements $a,b, \ldots$, be the set of *elementary actions.* For A we take an arbitrary, possibly infinite, set. Further, let *Stmv*, with typical elements $x,y, \ldots$, be the set of statement variables. For *Stmv* we take some infinite set of symbols.

DEFINITION 1.1 (Syntax for L_0)
We define the set of *statements* L_0, with typical elements $s,t, \ldots$, by the following syntax:

$$s ::= a \mid s_1;s_2 \mid s_1 \cup s_2 \mid s_1 \| s_2 \mid x \mid \mu x[t]$$

where $t \in L_0^x$, the set of statements which are *guarded for x,* to be defined below.

A statement s is of one of the following six forms:
- an *elementary action a.*
- the *sequential composition* $s_1;s_2$ of statements s_1 and s_2.
- the *nondeterministic choice* $s_1 \cup s_2$, also known as *local nondeterminism* [FHLR]: $s_1 \cup s_2$ is executed by executing either s_1 or s_2 chosen nondeterministically.
- the *concurrent execution* $s_1 \| s_2$, modeled by the arbitrary interleaving *(shuffle)* of the elementary actions of s_1 and s_2.
- a statement variable x, which is (normally) used in
- the recursive construct $\mu x[t]$: its execution amounts to execution of t where occurrences of x in t are executed by (recursively) executing $\mu x[t]$. For example, with the definition to be proposed presently, the intended meaning of $\mu x[(a;x) \cup b]$ is the set $a^* \cdot b \cup \{a^\omega\}$.

An important restriction of our language is that we consider only recursive constructs $\mu x[t]$, for which t is guarded for x: $t \in L_0^x$. Intuitively, a statement t is guarded for x when all occurrences of x in t are preceded by some statement. More formally:

DEFINITION 1.2 (Syntax for L_0^x)
The set L_0^x of statements which are guarded for x is given by

$$\begin{aligned} t ::= \; & a \\ \mid \; & t;s, \quad \text{for } s \in L_0 \\ \mid \; & t_1 \cup t_2 \mid t_1 \| t_2 \\ \mid \; & y \quad , \quad \text{for } y \neq x \\ \mid \; & \mu x[t] \end{aligned}$$

$$\mid \mu y[t'], \quad \text{for } y \neq x,\ t' \in L_0^x \cap L_0^y.$$

REMARK
In order to avoid possible confusion about the definitions of L_0 and L_0^x, let us give a more extensive definition, for which the ones given above are shorthand. We define L_0 and, for every $x \in Stmv$, L_0^x simultaneously and in stages:

Stage 0:

$$L_0(0) = A \cup Stmv, \quad L_0^x(0) = A \cup (Stmv \setminus \{x\})$$

Stage (n+1):

$$\begin{aligned} L_0(n+1) = L_0(n) &\cup \{s_1;s_2 \mid s_1,s_2 \in L_0(n)\} \\ &\cup \{s_1 \cup s_2 \mid s_1,s_2 \in L_0(n)\} \\ &\cup \{s_1 \| s_2 \mid s_1,s_2 \in L_0(n)\} \\ &\cup \{\mu x[t] \mid t \in L_0^x(n)\}. \end{aligned}$$

$$\begin{aligned} L_0^x(n+1) = L_0^x(n) &\cup \{t;s \mid t \in L_0^x(n),\ s \in L_0(n)\} \\ &\cup \{t_1 \cup t_2 \mid t_1,t_2 \in L_0^x(n)\} \\ &\cup \{t_1 \| t_2 \mid t_1,t_2 \in L_0^x(n)\} \\ &\cup \{\mu x[t] \mid t \in L_0^x(n)\} \\ &\cup \{\mu y[t] \mid y \neq x \wedge t \in L_0^x(n) \cap L_0^y(n)\}. \end{aligned}$$

We define

$$L_0 = \bigcup_{n \in \mathbb{N}} L_0(n), \quad L_0^x = \bigcup_{n \in \mathbb{N}} L_o^x(n).$$

REMARK (Empty statement)
It appears to be useful to have the languages under consideration contain a special element, denoted by E, which can be regarded as the empty statement. From now on E is considered to be an element of L_0, and L_0^x. We shall still write L_0 for $L_0 \cup \{E\}$ and L_0^x for $L_0^x \cup \{E\}$. Please note that syntactic constructs like $s;E$ or $E\|s$ are *not* in L_0.

Now that we have formulated the notion of guardedness for x, we can easily generalize this:

DEFINITION 1.3 (Guarded statements)
The set L_0^g of guarded statements (guarded for all x) is defined as

$$L_0^g = \cap_{x \in Stmv} L_0^x.$$

As L_0 and L_0^x, also L_0^g has a simple inductive structure.

LEMMA 1.4 *The set L_0^g can be given by the following syntax:*

$$t ::= a \mid t;s \mid t_1 \cup t_2 \mid t_1 \| t_2 \mid \mu x[t]$$

where $s \in L_0$.

We need yet another notion of syntactic nature, that is, the notion of closedness.

DEFINITION 1.5 (Free variables, closed statements)
For every statement $s \in L_0$ we define the set $FV(s)$ of all statement variables that occur freely in s as usual:

$$FV(a) = \emptyset,\ FV(x) = \{x\},\ FV(\mu x[s]) = FV(s) \setminus \{x\},$$
$$FV(s_1 ops_2) = FV(s_1) \cup FV(s_2), \text{ for } op = ;, \cup, \|.$$

We call a statement s *closed* (notation: closed (s)), whenever $FV(s) = \emptyset$. Finally, we define for $L = L_0$, L_0^x, and L_0^g:

$$L^{cl} = \{s \mid s \in L \mid \text{ closed } (s)\}$$

We have: $(L_0)^{cl} = (L_0^x)^{cl} = (L_0^g)^{cl}$.

We expect that the reader may benefit from a few

EXAMPLES
First we observe that $L_0^g \subseteq L_0^x \subseteq L_0$. Further we have that

$$x \in L_0,\ x \notin L_0^x$$
$$y;x \in L_0^x,\ y;x \notin L_0^y$$
$$\mu x[y;x] \in L_0,\ \mu y[y;x] \notin L_0$$
$$a;\mu x[y;x] \in L_0^x \cap L_0^y$$
$$\mu y[a;\mu x[y;x]] \in L_0$$

1.2 *Operational semantics*

We first introduce a semantic universe for both the operational and the denotational semantics for L_0.

DEFINITION 1.6 (Semantic universe P_0)
Let A^∞, the set of finite and infinite words over A, be given by

$$A^\infty = A^* \cup A^\omega.$$

For the empty word we use the special symbol ϵ. Let d_{A^∞} denote the usual metric on A^∞ (see example A.1.1). We define

$$P_0 = \mathcal{P}_{nc}(A^\infty),$$

with typical elements $p,q, \ldots$, the set of all non-empty, compact subsets of A^∞. As a metric on P_0 we take $d_{P_0} = (d_{A^\infty})_H$, the Hausdorff distance induced by d_{A^∞}. According to proposition A.7 we have that P_0 together with the metric d_{P_0} is a complete metric space.

The operational semantics for L_0 is based on the notion of a *transition relation.*

DEFINITION 1.7 (Transition relation for L_0^g)
We define a transition relation

$$\rightarrow \subseteq L_0^g \times A \times L_0$$

(writing $s \overset{a}{\rightarrow} s'$ for $(s,a,s') \in \rightarrow$) as the smallest relation satisfying

(i) $a \overset{a}{\rightarrow} E$ (for all $a \in A$)
(ii) for all $a \in A$, $s,t \in L_0^g$, $s',\overline{s} \in L_0$: if $s' \neq E$, then:

$$s \xrightarrow{a} s' \Rightarrow (s;\bar{s} \xrightarrow{a} s';\bar{s}$$
$$\wedge\ s \cup t \xrightarrow{a} s' \wedge t \cup s \xrightarrow{a} s'$$
$$\wedge\ s \| t \xrightarrow{a} s' \| t \wedge t \| s \xrightarrow{a} t \| s'$$
$$\wedge\ \mu x[s] \xrightarrow{a} s'[\mu x[s]/x]),$$

where the latter statement is obtained by replacing all free occurrences of x in s by $\mu x[s]$; and if $s' = E$, then:

$$s \xrightarrow{a} E \Rightarrow (s;\bar{s} \xrightarrow{a} \bar{s}$$
$$\wedge\ s \cup t \xrightarrow{a} E \wedge t \cup s \xrightarrow{a} E$$
$$\wedge\ s \| t \xrightarrow{a} t \wedge t \| s \xrightarrow{a} t$$
$$\wedge\ \mu x[s] \xrightarrow{a} E).$$

Intuitively, $s \xrightarrow{a} s'$ tells us that s can do the elementary action a as a first step, resulting in the statement s'. We now give the definition of $\mathcal{O}_0$, the operational semantics for L_0^{cl}. (It is defined on *closed* statements only, because we do not want to give an operational meaning to, e.g., $a;x$: what should it be?) It will be the fixed point of the following contraction.

DEFINITION 1.8 (Φ_0)
Let $\Phi_0 : (L_0^{cl} \to P_0) \to (L_0^{cl} \to P_0)$ be given by

$$\Phi_0(F)(s) = \begin{cases} \{\epsilon\} & \text{if } s = E \\ \bigcup \{a \cdot F(s') \mid s' \in L_0^{cl} \wedge a \in A \wedge s \xrightarrow{a} s'\} & \text{if } s \neq E \end{cases}$$

for $F \in L_0^{cl} \to P_0$ and $s \in L_0^{cl}$.

REMARKS 1.9
(1) It is straightforward to prove that Φ_0 is contracting.
(2) Please note that closed (s) and $s \xrightarrow{a} s'$ imply closed (s').
(3) We have that $\Phi_0(F)(s)$ is a non-empty, compact subset of A^∞, because $\{a \mid \exists s' \in L_0^{cl}[s \xrightarrow{a} s']\}$ is finite and non-empty (this follows from lemma 1.14 below) and $F(s')$ is compact for every $s' \in L_0^{cl}$. This implies that $\Phi_0(F) \in L_0^{cl} \to P_0$.

DEFINITION 1.10 ($\mathcal{O}_0$): $\mathcal{O}_0 = \text{Fixed Point}(\Phi_0)$

REMARK: We use open brackets to denote application of $\mathcal{O}_0$ to an argument s: $\mathcal{O}_0[\![s]\!]$.

In [BKMOZ] another, seemingly more operational, definition of $\mathcal{O}_0$ is given. We shall repeat a slightly different version of it here and show that it is equivalent to this fixed-point definition.

DEFINITION 1.11 ($\mathcal{O}_0^*$)
Let $s \in L_0^{cl}$, $s \neq E$. We define $\mathcal{O}_0^* : L_0^{cl} \to P_0$ by putting $w \in A^\infty$ in $\mathcal{O}_0^*[\![s]\!]$ if and only if one of the following two conditions is satisfied:

(*i*) $s \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \xrightarrow{a_3} \cdots \xrightarrow{a_n} s_n \wedge s_n = E \wedge w = a_1 \cdots a_n$

$$(ii)\quad s \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \xrightarrow{a_3} \cdots \xrightarrow{a_n} s_n \xrightarrow{a_{n+1}} \cdots \ \wedge\ w = a_1 \cdots a_n a_{n+1} \cdots$$

(where $s \xrightarrow{a_1} s' \xrightarrow{a_2} s''$ abbreviates $s \xrightarrow{a_1} s' \wedge s' \xrightarrow{a_2} s''$). If $s = E$, then $\mathcal{O}_0^*[\![s]\!] = \{\epsilon\}$.

LEMMA 1.12: $\mathcal{O}_0 = \mathcal{O}_0^*$

PROOF
Let $w \in A^\infty$, $s \in L_0^{cl}$, with $s \neq E$. We have

$$w \in \mathcal{O}_0^*[\![s]\!] \Leftrightarrow \text{[definition } \mathcal{O}_0^*\text{]}$$

$$\exists a \in A\, \exists s' \in L_0^{cl}\, \exists w' \in A^\infty\ [s \xrightarrow{a} s' \wedge w = a \cdot w' \wedge w' \in \mathcal{O}_0^*[\![s]\!]]$$

$$\Leftrightarrow \text{[definition } \Phi_0\text{]}$$

$$w \in \Phi_0(\mathcal{O}_0^*)(s).$$

Since $\mathcal{O}_0^* \in L_0^{cl} \to P_0$, it follows that $\mathcal{O}_0^* = \Phi(\mathcal{O}_0^*)$. Thus $\mathcal{O}_0^* = \mathcal{O}_0$.

We give yet another characterization of $\mathcal{O}_0$. It is based on the following definition and will be the one we use in proving semantic equivalence.

DEFINITION 1.13 (Initial steps)
We define a function

$$I : L_0^g \to \mathcal{P}_{fin}(A \times L_0)$$

(where $\mathcal{P}_{fin}(X) = \{Y \mid Y \subseteq X \wedge \text{ finite } (Y)\}$) by induction on L_0^g:

(i) $I(E) = \varnothing$, and $I(a) = \{(a,\ E)\}$

(ii) Suppose $I(s) = \{(a_i,\ s_i)\}$, $I(t) = \{(b_j,\ t_j)\}$ for $s, t \in L_0^g$, $a_i, b_j \in A$, $s_i, t_j \in L_0$. (The variables i and j range over some finite sets of indices, which we have omitted.) Then

$$I(s\,;\overline{s}) = \{(a_i,\ s_i;\overline{s})\} \text{ (for } \overline{s} \in L_0)$$

$$I(s \cup t) = I(s) \cup I(t)$$

$$I(s \| t) = \{(a_i,\ s_i \| t)\} \cup \{(b_j,\ s \| t_j)\}$$

$$I(\mu x[s]) = \{(a_i,\ s_i[\mu x[s]/x])\}.$$

REMARK: Please note that for all $s \neq E$ the set $I(s)$ is finite and non-empty.

This definition is motivated by the following lemma, which can be easily proved.

LEMMA 1.14: $\forall a \in A\, \forall s \in L_0^g\, \forall s' \in L_0\ [s \xrightarrow{a} s' \Leftrightarrow (a,\ s') \in I(s)]$

COROLLARY 1.15: $\Phi_0(F)(s) = \bigcup \{a \cdot F(s') \mid (a, s') \in I(s)\}$, for $F : L_0^{cl} \to P_0$, $s \in L_0^{cl} \setminus \{E\}$.

1.3 *Denotational semantics*

The second semantic function we define for L_0 will be *denotational:* We call a semantic function $F : L_0 \to M$ (where M is some mathematical domain) denotational if it is compositionally defined and tackles recursion with the help of fixed points. The first condition is satisfied if for every syntactic operator *op* in L_0 we can define a corresponding semantic operator $\tilde{op} : M \times M \to M$ (assuming *op* to be binary) such that

$$F(s_1 ops_2) = F(s_1)\tilde{op}F(s_2).$$

As semantic domain for the denotational semantics of L_0 we take again P_0. The semantic operators corresponding with ; , $\cup$, and $\|$, the syntactic operators in L_0, will be of type $P_0 \times P_0 \to P_0$.

DEFINITION 1.16 (Semantic operators)
The operators $\tilde{;}$, $\tilde{\cup}$, $\tilde{\|}$: $P_0 \times P_0 \to P_0$ are defined as follows. Let $p, q \in P_0$, then

$$(i) \quad p\tilde{;}q = \begin{cases} q & \text{if } p = \{\epsilon\} \\ \bigcup \{a \cdot (p_a \tilde{;} q) \mid p_a \neq \emptyset\} & \text{otherwise} \end{cases}$$

$$(ii) \quad p \tilde{\cup} q = p \cup q \quad \text{(set-theoretic union)}$$

$$(iii) \quad p\tilde{\|}q = \begin{cases} p & \text{if } q = \{\epsilon\} \\ q & \text{if } p = \{\epsilon\} \\ \bigcup \{a \cdot (p_a \tilde{\|} q) \mid p_a \neq \emptyset\} \cup \bigcup \{a \cdot (p \tilde{\|} q_a) \mid q_a \neq \emptyset\} & \text{otherwise,} \end{cases}$$

where for every $p \in P_0$ and $a \in A$ we define:

$$p_a = \{w \mid w \in A^\infty \wedge a \cdot w \in p\}.$$

(We often write op rather than $\tilde{op}$ if no confusion is possible.)

REMARKS 1.17
(1) This definition is self-referential and needs some justification. We shall give it for $\tilde{;}$ and leave the case of $\tilde{\|}$ to the reader. We define a mapping: $\Phi: (P_0 \times P_0 \to P_0) \to (P_0 \times P_0 \to P_0)$ by

$$\Phi(F)(p,q) = \begin{cases} q & \text{if } p = \{\epsilon\} \\ \bigcup \{a \cdot F(p_a, q) \mid p_a \neq \emptyset\} & \text{otherwise.} \end{cases}$$

It is not difficult to show that Φ is contracting. Then we define: $\tilde{;} = \text{Fixed Point}(\Phi)$, which satisfies the equation of definition 1.16 above.

(2) If we define the *left-merge* operator $\tilde{\llcorner}$ by

$$p\tilde{\llcorner}q = \begin{cases} \emptyset & \text{if } p = \{\epsilon\} \\ \bigcup \{a \cdot (p_a \| q) \mid p_a \neq \emptyset\} & \text{otherwise,} \end{cases}$$

then we have that

$$p \| q = p\tilde{\llcorner}q \cup q\tilde{\llcorner}p$$

(using the fact that $p' \| q' = q' \| p'$, for all p' and q'). This abbreviation will be helpful in some future proofs.

We need the following properties, which are easily verified:

LEMMA 1.18
(a) For $op = \tilde{;}$, $\tilde{\cup}$, and $\tilde{\|}$ we have

$$\forall p,p',q,q' \in P_0 \; [d_{P_0}(p \; op \; q, p' \; op \; q') \leqslant \max\{d_{P_0}(p,p'), d_{P_0}(q,q')\}].$$

(b) For $p,p' \in P_0$ with $\epsilon \notin p$, $\epsilon \notin p'$, and $q,q' \in P_0$ we have

$$d_{P_0}(p\tilde{;}q, p'\tilde{;}q') = \max\{d_{P_0}(p,p'), \tfrac{1}{2} \cdot d_{P_0}(q,q')\}.$$

(c) The operators $\tilde{;}$, $\tilde{\cup}$, and $\tilde{\|}$ preserve compactness.

We shall treat recursion with the help of environments, which are used to store and retrieve meanings of statement variables. They are defined in

DEFINITION 1.19 (Semantic environments)
The set Γ of *semantic environments*, with typical elements γ, is given by

$$\Gamma = Stmv \to^{fin} P_0.$$

We write $\gamma\{p/x\}$ for a *variant* of γ which is like γ but with $\gamma\{p/x\}(x)=p$.

Now we have defined everything we need to introduce the denotational semantics for L_0.

DEFINITION 1.20 (Ψ_0, D_0)
We shall define D_0 as the fixed point of

$$\Psi_0 : (L_0 \to \Gamma \to {}^1P_0) \to (L_0 \to \Gamma \to {}^1P_0)$$

which is given by induction on L_0. (Here $\Gamma \to {}^1P_0$ denotes the set of non-distance-increasing functions (see A.3.(c)).) Let $F \in L_0 \to \Gamma \to {}^1P_0$, then:

$$(i) \quad \Psi_0(F)(a)(\gamma)=\{a\},\ \Psi_0(F)(x)(\gamma)=\gamma(x),\ \Psi_0(F)(E)(\gamma)=\{\epsilon\}$$

$$(ii) \quad \Psi_0(F)(s\ op\ t)(\gamma) = \Psi_0(F)(s)(\gamma)\ \tilde{op}\ \Psi_0(F)(t)(\gamma)$$

$$(iii) \quad \Psi_0(F)(\mu x[s])(\gamma) = \Psi_0(F)(s)(\gamma\{F(\mu x[s])(\gamma)/x\}) \text{ for } s \in L_0^x,$$

for $op = ;, \cup, \|$, and $\tilde{op}$ as in definition 1.16. (We define $\Psi_0(F)$ only for those s and γ, such that $FV(s) \subseteq dom(\gamma)$.) Now we set

$$D_0 = \text{Fixed Point}(\Psi_0).$$

REMARK: We have: $D_0[\![\mu x[s]]\!](\gamma) = D_0[\![s]\!](\gamma\{D_0[\![\mu x[s]]\!](\gamma)/x\})$. (As for $\mathcal{O}_0$, we also use open brackets for $\mathcal{D}_0$.)

It is not obvious that Ψ_0 is contracting. The fact that we consider only *guarded* recursion is essential for proving it.

LEMMA 1.21
(a) *If* $F \in L_0 \to \Gamma \to {}^1P_0$, *then* $\Psi_0(F) \in L_0 \to \Gamma \to {}^1P_0$.
(b) *If* $F \in L_0 \to \Gamma \to {}^1P_0$, *then for all* $\gamma_1, \gamma_2 \in \Gamma, s \in L_0$:
(*) $\forall y \in Stmv[s \notin L_0^y \Rightarrow \gamma_1(y)=\gamma_2(y)]$
$\Rightarrow$
(**) $d_{P_0}(\Psi_0(F)(s)(\gamma_1), \Psi_0(F)(s)(\gamma_2)) \leqslant \frac{1}{2} \cdot d_\Gamma(\gamma_1, \gamma_2)$.
(c) Ψ_0 *is contracting on* $L_0 \to \Gamma \to {}^1P_0$.

PROOF
(a) The proof of (a) goes along the lines of (b), which is more interesting.
(b) Let $F \in L_0 \to {}^1P_0$, let $\gamma_1, \gamma_2 \in \Gamma$. We use induction on L_0.
(i) For $s=a$ we have: $d_{P_0}(\Psi_0(F)(a)(\gamma_1), \Psi_0(F)(a)(\gamma_2))=0$. Let $s=x$, with $x \in Stmv$. Suppose (*) holds for x. Then

$$d_{P_0}(\Psi_0(F)(x)(\gamma_1), \Psi_0(F)(x)(\gamma_2)) = d_{P_0}(\gamma_1(x), \gamma_2(x))$$
$$= 0 \text{ [because of (*)].}$$

(ii) We only treat sequential composition and recursion. Let $s=s_1;s_2$, with $s_1, s_2 \in L_0$. Suppose (b)

holds for s_1 and s_2. Suppose (*) holds for $s_1;s_2$. This implies that (*) holds for s_1. Thus we have (**) for s_1. Now:

$$
\begin{aligned}
& d_{P_0}(\Psi_0(F)(s_1;s_2)(\gamma_1), \Psi_0(F)(s_1;s_2)(\gamma_2)) \\
& = d_{P_0}(\Psi_0(F)(s_1)(\gamma_1)\tilde{;}\Psi_0(F)(s_2)(\gamma_1), \Psi_0(F)(s_1)(\gamma_2)\tilde{;}\Psi_0(F)(s_2)(\gamma_2)) \\
& \leqslant [\text{for all } s\in L_0\setminus\{E\},\ F \text{ and } \gamma \text{ we have: } \epsilon\notin\Psi_0(F)(s)(\gamma);\ \text{thus lemma 1.18}(b)\ \text{applies}] \\
& \quad \max\{d_{P_0}(\Psi_0(F)(s_1)(\gamma_1), \Psi_0(F)(s_1)(\gamma_2)), \tfrac{1}{2}\cdot d_{P_0}(\Psi_0(F)(s_2)(\gamma_1), \Psi_0(F)(s_2)(\gamma_2))\} \\
& \leqslant [(**) \text{ for } s_1;\ (a) \text{ for } s_2] \\
& \quad \max\{\tfrac{1}{2}\cdot d_\Gamma(\gamma_1,\gamma_2), \tfrac{1}{2}\cdot d_\Gamma(\gamma_1,\gamma_2)\} \\
& = \tfrac{1}{2}\cdot d_\Gamma(\gamma_1,\gamma_2).
\end{aligned}
$$

(The proof for $s_1 \cup s_2$ and $s_1 \| s_2$ is similar.) Next we treat recursion. Let $s_1\in L_0$ and suppose that $\mu x[s_1]$ satisfies (*). Then s_1 satisfies it. Thus we have (**) for s_1. Now

$$
\begin{aligned}
& d_{P_0}(\Psi_0(F)(\mu x[s_1])(\gamma_1), \Psi_0(F)(\mu x[s_1])(\gamma_2)) \\
& = d_{P_0}(\Psi_0(F)(s)(\gamma_1\{F(\mu x[s_1])(\gamma_1)/x\}), \Psi_0(F)(s)(\gamma_2\{F(\mu x[s_1])(\gamma_2)/x\})) \\
& \leqslant [(*) \text{ holds for } s_1, \text{ also w.r.t. } \gamma_i\{F(\mu x[s_1])(\gamma_i)/x\}, \text{ for } i = 1,2, \text{ thus so does}(**)] \\
& \quad \tfrac{1}{2}\cdot d_\Gamma(\gamma_1\{F(\mu x[s_1])(\gamma_1)/x\}, \gamma_2\{F(\mu x[s_1])(\gamma_2)/x\}) \\
& \leqslant \tfrac{1}{2}\cdot\max\{d_\Gamma(\gamma_1,\gamma_2), d_{P_0}(F(\mu x[s_1])(\gamma_1), F(\mu x[s_1])(\gamma_2))\} \\
& \leqslant [(a) \text{ for } \mu x[s_1]] \\
& \quad \tfrac{1}{2}\cdot d_\Gamma(\gamma_1,\gamma_2).
\end{aligned}
$$

(c) Let $F_1, F_2\in L_0\to\Gamma\to{}^1P_0$. We only treat recursion. Suppose $d_{P_0}(\Psi_0(F_1)(s)(\gamma), \Psi_0(F_2)(s)(\gamma)) \leqslant\tfrac{1}{2}\cdot d(F_1,F_2)$, for some $s\in L_0^{\times}$, all $\gamma\in\Gamma$. Then

$$
\begin{aligned}
& d_{P_0}(\Psi_0(F_1)(\mu x[s])(\gamma), \Psi_0(F_2)(\mu x[s])(\gamma)) \\
& = [\gamma_i = \gamma\{F_i(\mu x[s])(\gamma)/x\},\ i = 1,2] \\
& \quad d_{P_0}(\Psi_0(F_1)(s)(\gamma_1), \Psi_0(F_2)(s)(\gamma_2)) \\
& \leqslant \max\{d_{P_0}(\Psi_0(F_1)(s)(\gamma_1), \Psi_0(F_2)(s)(\gamma_1)), d_{P_0}(\Psi_0(F_2)(s)(\gamma_1), \Psi_0(F_2)(s)(\gamma_2))\} \\
& \leqslant [\text{induction, } (b)] \\
& \quad \max\{\tfrac{1}{2}\cdot d(F_1,F_2), \tfrac{1}{2}\cdot d_\Gamma(\gamma_1,\gamma_2)\} \\
& = \max\{\tfrac{1}{2}\cdot d(F_1,F_2), \tfrac{1}{2}\cdot d_{P_0}(F_1(\mu x[s])(\gamma), F_2(\mu x[s])(\gamma))\} \\
& = \tfrac{1}{2}\cdot d(F_1,F_2).
\end{aligned}
$$

1.4 *Semantic equivalence of* $\mathcal{O}_0$ and $\mathcal{D}_0$

An important difference between $\mathcal{D}_0$ and $\mathcal{O}_0$ is that recursion is treated with and without semantic environments, respectively. We have

$$\mathcal{D}_0[\![\mu x[s]]\!](\gamma) = \mathcal{D}_0[\![s]]\!](\gamma\{\mathcal{D}_0[\![\mu x[s]]\!](\gamma)/x\})$$

and

$$\mathcal{O}_0[\![\mu x[s]]\!] = \mathcal{O}_0[\![s[\mu x[s]/x]]\!].$$

In the latter case the statement $\mu x[s]$ is *syntactically* substituted for all free statement variables x in s, whereas in the first case the environment γ is changed by setting x to the *semantic* value of $\mu x[s]$. We shall compare $\mathcal{O}_0$ and $\mathcal{D}_0$ by relating both to an intermediate semantic function $\mathcal{O}_0'$, which takes *syntactic* instead of *semantic* environments as arguments. It will be defined such that for syntactic environments δ:

$$\mathcal{O}_0'[\![\mu x[s]]\!](\delta) = \mathcal{O}_0'[\![s]\!](\delta\{\mu x[s]/x\}).$$

Here δ is changed, the new value of x is the statement $\mu x[s]$. By first comparing $\mathcal{O}_0$ and $\mathcal{O}_0'$ and next $\mathcal{O}_0'$ and $\mathcal{D}_0$ we are able to prove the main result of this section: $\mathcal{O}_0[\![s]\!] = D_0[\![s]\!](\gamma)$, for all $s \in L_0^{cl}$ and arbitrary $\gamma \in \Gamma$. For the definition of $\mathcal{O}_0'$, we need

DEFINITION 1.22 (Syntactic environments)
The set Δ of *syntactic* environments, with typical elements δ, is defined by

$$\Delta = \{\delta \mid \delta \in (Stmv \to^{\text{fin}} L_0) \wedge (\delta \text{ is normal})\},$$

where the notion of *normal* environments is given in:

DEFINITION 1.23 (Normal environments)
A syntactic environment δ is called *normal*, whenever

(i) $\forall x \in dom(\delta)\ [\delta(x) \in L_0^x]$

(ii) $\forall s \in L_0\ [FV(s) \subseteq dom(\delta) \Rightarrow \exists k \geqslant 0\ [s[\delta]^k \in L_0^{cl}]]$,

where $s[\delta]^0 = s$, $s[\delta]^1 = s[\delta(x_1)/x_1, \ldots, \delta(x_n)/x_n]$ (with $FV(s) = \{x_1, \ldots, x_n\}$) and $s[\delta]^{n+1} = (s[\delta])[\delta]^n$. For δ normal and $s \in L_0$, with $FV(s) \subseteq dom(\delta)$, we define

$$s\langle\delta\rangle = s[\delta]^k,$$

where $k = \min\{m \mid s[\delta]^m \in L_0^{cl}\}$.

REMARKS
(1) From now on we shall assume whenever we consider $s \in L_0$ and $\delta \in \Delta$ together (as two arguments for a function, or as a pair) that $FV(s) \subseteq dom(\delta)$.
(2) Let $\delta \in Stmv \to^{fin} L_0$ be such that for $x,y \in Stmv$: $\delta(x) = y$ and $\delta(y) = x$. Such an environment is *not* normal. It does not give us any useful information about the values of x and y.
(3) It would be too restrictive to require for all $\delta \in Stmv \to^{fin} L_0$ that $\forall x \in dom(\delta)\ [x[\delta] \in L_0^{cl}]$. An example may illustrate this. Let δ be defined such that $dom(\delta) = \{x,y\}$, and

$$\delta(y) = \mu y[b;x;y],\quad \delta(x) = \mu x[a;\mu y[b;x;y]].$$

Such an environment we shall encounter when computing $\mathcal{O}_0'[\![\mu x[a;\mu y[b;x;y]]]\!]$. Now $y[\delta] = \delta(y) \notin L_0^{cl}$, but $y[\delta]^2 \in L_0^{cl}$.

Now that we have introduced syntactic environments, we can formulate a principle of induction for the set $L_0 \times \Delta$, which we shall heavily use in the sequel.

THEOREM 1.24 (Induction principle for $L_0 \times \Delta$)
Let $\Xi \subseteq L_0 \times \Delta$. *If:*

(1) $A \times \Delta \subseteq \Xi$

(2) $\{s,t\} \times \Delta \subseteq \Xi \Rightarrow \{s;\bar{s}, s \cup t, s \| t\} \times \Delta \subseteq \Xi$ *for* $s,t,\bar{s} \in L_0$

(3) $\{s\}\times\Delta\subseteq\Xi \Rightarrow \{\mu x[s]\}\times\Delta\subseteq\Xi$ for $s\in L_0^x$

(4) $(\delta(x),\delta)\in\Xi \Rightarrow (x,\delta)\in\Xi$ *for* $x\in Stmv$ *and* $\delta\in\Delta$,

then:

$$\Xi = L_0\times\Delta.$$

PROOF

Let $\Xi\subseteq L_0\times\Delta$, suppose Ξ satisfies (1) through (4). We first prove fact (a) and fact (b) given below, and next show that (a) and (b) imply: $\Xi=L_0\times\Delta$. So we have

fact (a): $L_0^g\times\Delta\subseteq\Xi$

fact (b): $\forall S\subseteq L_0\times\Delta\,[S\subseteq\Xi\Rightarrow S'\subseteq\Xi]$, where

$$S' = \{(s,\delta)\,|\,(s,\delta)\in L_0\times\Delta\wedge\forall x\in FV(s)\,[s\notin L_0^x \Rightarrow (\delta(x),\,\delta)\in S]\}.$$

To show that (a) holds, we use (1), (2), and (3), and induction on the structure of L_0^g. We proceed with (b). Let $S\subseteq L_0\times\Delta$ and suppose $S\subseteq\Xi$. Let S' be as above. We use (1) through (4) and induction on L_0 to show that $S'\subseteq\Xi$. Let $(s,\delta)\in S'$, for $s\in L_0$, $\delta\in\Delta$.

(i) $s\equiv a$: $(a,\delta)\in\Xi$, because (1).

(ii) $s\equiv s_1 ops_2$: Suppose that if $(s_i,\delta)\in S'$, then $(s_i,\delta)\in\Xi$, for $i=1,2$. If $(s,\delta)\in S'$, then also (s_1,δ) and $(s_2,\delta)\in S'$. Thus $(s_1,\delta),(s_2,\delta)\in\Xi$. By (2) we have: $(s_1\ op\ s_2,\delta)\in\Xi$.

(iii) $s\equiv\mu x[s_1]$, for $s_1\in L_0^x$: Suppose that $(s_1,\delta)\in S'$ implies $(s_1,\delta)\in\Xi$. Because $s_1\in L_0^x$ we have: $(s_1,\delta)\in S'\Leftrightarrow(\mu x[s_1],\delta)\in S'$. Because $(\mu x[s_1],\delta)\in S'$ we have $(s_1,\delta)\in\Xi$. Thus, using (3), we have $(\mu x[s_1],\delta)\in\Xi$.

(iv) $s\equiv x$: If $(x,\delta)\in S'$, then $(\delta(x),\delta)\in S$, thus (because $S\subseteq\Xi$) $(\delta(x),\delta)\in\Xi$. Because of (4), we then have that $(x,\delta)\in\Xi$.

Thus facts (a) and (b) hold. Next we show that $\Xi=L_0\times\Delta$. For this purpose we define, for all $n\in\mathbb{N}$:

$$V_0 = L_0^g\times\Delta,$$

$$V_{n+1} = \{(s,\delta)\,|\,(s,\delta)\in L_0\times\Delta\wedge\forall x\in FV(s)\,[s\notin L_0^x \Rightarrow (\delta(x),\,\delta)\in V_n]\}.$$

Then we have:

(*) $\forall s\in L_0\forall\delta\in\Delta\exists n\in\mathbb{N}\,[s[\delta]^n\in L_0^g\Rightarrow(s,\delta)\in V_n]$,

which we prove with induction on $n\in\mathbb{N}$. Let $s\in L_0$ and $\delta\in\Delta$. If $s[\delta]^0\in L_0^g$, then $s\in L_0^g\subseteq L_0^g$. Thus $(s,\delta)\in V_0$. Now suppose (*) holds for $n\in\mathbb{N}$, and suppose $s[\delta]^{n+1}\in L_0^g$. Then $(s[\delta])[\delta]^n\in L_0^g$, thus by induction $(s[\delta],\delta)\in V_n$. This implies $(s,\delta)\in V_{n+1}$, which proves (*) for $n+1$.

Because all $\delta\in\Delta$ are normal we have

$$\forall(s,\delta)\in L_0\times\Delta\exists n\in\mathbb{N}\,[s[\delta]^n\in L_0^g].$$

Together with (*) this implies:

$$\forall(s,\delta)\in L_0\times\Delta\exists n\in\mathbb{N}\,[(s,\delta)\in V_n].$$

Since $V_n\subseteq L_0\times\Delta$, for all $n\in\mathbb{N}$, it follows that $L_0\times\Delta=\cup_{n\in\mathbb{N}}V_n$. Now $V_0\subseteq\Xi$ because of (a), and $V_n\subseteq\Xi\Rightarrow V_{n+1}\subseteq\Xi$ because of (b), so we conclude: $\Xi=L_0\times\Delta$.

□

REMARK

We cannot reason about a free statement variable x unless we know what statement it is bound to. Therefore, we consider non-closed statements together with syntactic environments, which give information about the free variables they contain. This explains why we have formulated an induction principle for $L_0\times\Delta$ instead of L_0 only.

Now let $\Xi \subseteq L_0 \times \Delta$. The first three conditions of the principle suffice to prove that $L_0^g \times \Delta \subseteq \Xi$, since they express exactly the syntactic structure of L_0^g (see lemma 1.4). (We have chosen L_0^g here instead of L_0^c, because the latter set has no simple inductive structure.) Thus also $L_0^c \times \Delta$ ($\subseteq L_0^g \times \Delta$) $\subseteq \Xi$. Adding condition (4) enables us to prove $L_0 \times \Delta \subseteq \Xi$. This may be motivated by the following. For every statement $s \in L_0$ and normal environment $\delta \in \Delta$ there exists an $l \in \mathbb{N}$ such that $s[\delta]^l \in L_0^c \subseteq L_0^g$. Let us call $k \in \mathbb{N}$ with $k = \min\{l \mid s[\delta]^l \in L_0^c\}$ the *degree of closedness* of s with respect to δ. Please note that every $s \in L_0^c$ has degree 0, and arbitrary $s \in L_0$ has, for arbitrary δ, a finite degree. Therefore, this degree can be used as a measure for the complexity of statements. Our induction principle is indeed a principle of induction on the degree of closedness. Conditions (1), (2), and (3) are sufficient to prove Ξ for all (s, δ) with degree 0. They form, so to speak, the basis of the principle. Condition (4) expresses the "step part": if Ξ holds for $(\delta(x), \delta)$, which has degree k, say, then Ξ holds for (x, δ), which then has degree $k+1$.

We now proceed with the definition of $\mathcal{O}_0'$. It will be of type

$$\mathcal{O}_0' : L_0 \to \Delta \to P_0,$$

which could be called intermediate between

$$\mathcal{O}_0 : L_0^c \to P_0, \text{ and } D_0 : L_0 \to \Gamma \to P_0.$$

Instead of basing the definition of $\mathcal{O}_0'$ on some transition relation (as in definition 1.8) we use a variant of the initial step function (definition 1.13).

DEFINITION 1.25 (Initial steps with syntactic environments)
We define a function

$$I' : L_0 \to \Delta \to \mathcal{P}_{fin}(A \times L_0 \times \Delta),$$

using the induction principle for $L_0 \times \Delta$. The predicate $\Xi \subseteq L_0 \times \Delta$ we use is defined as:

$$\Xi(s, \delta) \equiv I'(s)(\delta) \text{ is defined.}$$

We shall define I' such that Ξ satisfies the induction conditions. Thus we ensure that I' is defined for every $s \in L_0$ and $\delta \in \Delta$ (with $FV(s) \subseteq dom(\delta)$).

(1) $I'(E)(\delta) = \varnothing$, and $I'(a)(\delta) = \{(a, E, \delta)\}$, for all $a \in A$, $\delta \in \Delta$.

(2) Suppose $I'(s) = \lambda\delta \cdot \{(a_i, s_i, \delta_i)\}$, $I'(t) = \lambda\delta \cdot \{(b_j, t_j, \delta_j)\}$ for $s, t, s_i, t_j \in L_0$, $a_i, b_j \in A$, and $\delta_i, \delta_j \in \Delta$. (The variables i and j range over some finite sets of indices, which are omitted.) Then:

$$I'(s\,;\overline{s})(\delta) = \{(a_i,\ s_i;\overline{s},\ \delta_i)\} \quad (\textit{for } \overline{s} \in L_0)$$

$$I'(s \cup t)(\delta) = I'(s)(\delta) \cup I'(t)(\delta)$$

$$I'(s \| t)(\delta) = \{(a_i,\ s_i \| t,\ \delta_i)\} \cup \{(b_j,\ s \| t_j,\ \delta_j)\}$$

(3) For the definition of $I'(\mu x[s])$ we have to consider possible clashes of variables. Therefore, we distinguish between two cases (supposing that $I'(s)$ has already been defined):

$$I'(\mu x[s])(\delta) = \begin{cases} I'(s)(\delta\{\mu x[s]/x\}) & \text{if } x \notin dom(\delta) \\ I'(\overline{s})(\delta\{\mu\overline{x}[\overline{s}]/\overline{x}\}) & \text{if } x \in dom(\delta), \end{cases}$$

where $\overline{x}$ is some fresh variable with $\overline{x} \notin dom(\delta)$ and $\overline{s} = s[\overline{x}/x]$.

(4) Suppose $I'(\delta(x))(\delta)$ has already been defined. We set:

$$I'(x)(\delta) = I'(\delta(x))(\delta).$$

REMARKS

(1) We have: if $I'(s)(\delta) = \{(a_i, s_i, \delta_i)\}$, then *normal* (δ_i), and thus $\delta_i \in \Delta$, for all i.

(2) The definition of $I'(\mu x[s])(\delta)$, with $x \in dom(\delta)$, is correct, because s and $\overline{s}$ have the same complexity.
(3) If $I'(s)(\delta) = \{(a_i, s_i, \delta_i)\}$, then for all i: $\forall x \in Stmv[x \in dom(\delta) \cap dom(\delta_i) \Rightarrow \delta(x) = \delta_i(x)]$.

DEFINITION 1.26 (Φ_0')
We define $\Phi_0' : (L_0 \to \Delta \to P_0) \to (L_0 \to \Delta \to P_0)$ by

$$\Phi_0'(F)(s)(\delta) = \begin{cases} \{\epsilon\} & \text{if } s = E \\ \bigcup \{a \cdot F(s')(\delta') \mid (a, s', \delta') \in I'(s)(\delta)\} & \text{otherwise} \end{cases}$$

for $F \in L_0 \to \Delta \to P_0$, $s \in L_0$, and $\delta \in \Delta$ with $FV(s) \subseteq dom(\delta)$.

DEFINITION 1.27: $\mathcal{O}_0' = \text{Fixed Point}(\Phi_0')$

Next, we compare $\mathcal{O}_0$ and $\mathcal{O}_0'$. We can do this by relating I and I', since we have:

$$\mathcal{O}_0[\![s]\!] = \bigcup \{a \cdot \mathcal{O}_0[\![s']\!] \mid (a, s') \in I(s)\}, \text{ for } s \in L_0^d,\ s \neq E$$

$$\mathcal{O}_0'[\![s]\!](\delta) = \bigcup \{a \cdot \mathcal{O}_0'[\![s']\!](\delta') \mid (a, s', \delta') \in I'(s)(\delta)\}, \text{ for } s \in L_0,\ s \neq E,\ \delta \in \Delta.$$

THEOREM 1.28 (Relating I and I')
For all $s \in L_0$ and $\delta \in \Delta$, with $FV(s) \subseteq dom(\delta)$, we have:

$$\forall a \in A \forall s' \in L_0 \forall \delta' \in \Delta\ [(a, s', \delta') \in I'(s)(\delta) \Leftrightarrow (a, s' \langle \delta' \rangle) \in I(s \langle \delta \rangle)].$$

(For the definition of $s \langle \delta \rangle$ see 1.23.)

PROOF
We define

$$\Xi(s, \delta) \equiv \forall a \in A \forall s' \in L_0 \forall \delta' \in \Delta\ [(a, s', \delta') \in I'(s)(\delta) \Leftrightarrow (a, s' \langle \delta' \rangle) \in I(s \langle \delta \rangle)]$$

and use the induction principle for $L_0 \times \Delta$ to show that $\Xi = L_0 \times \Delta$. We only treat the case of recursion. Suppose $s \in L_0^x$ such that $\{s\} \times \Delta \subseteq \Xi$. We have to show that $\{\mu x[s]\} \times \Delta \subseteq \Xi$. Let $\delta \in \Delta$ and assume (without loss of generality) that $x \notin dom(\delta)$. Then

$$I'(\mu x[s])(\delta) = I'(s)(\delta')$$

where $\delta' = \delta\{\mu x[s]/x\}$ (by the definition of I'). On the other hand, we have

$$\begin{aligned} I(\mu x[s] \langle \delta \rangle) &= [x \notin dom(\delta)] \\ &\quad I(\mu x[s \langle \delta \rangle]) \\ &= I(s \langle \delta \rangle [\mu x[s \langle \delta \rangle]/x]) \end{aligned}$$

(the latter equality following from:

$$\forall t \in L_0^{\delta}\ [I(\mu x[t]) = I(t[\mu x[t]/x])].$$

We take a quick (but deep) breath and proceed as follows:

$$\begin{aligned} s \langle \delta \rangle [\mu x[s \langle \delta \rangle]/x] &= [\text{definition } s \langle \delta \rangle] \\ &\quad s[\delta] \langle \delta \rangle [\mu x[s \langle \delta \rangle]/x] \\ &= [x \notin dom(\delta),\ \forall y \in dom(\delta)\ [x \notin FV(\delta(y))]] \\ &\quad s[\delta][\mu x[s \langle \delta \rangle]/x] \langle \delta \rangle \\ &= s[\delta][\mu x[s]/x] \langle \delta \rangle \end{aligned}$$

$$= [\delta' = \delta\{\mu x[s]/x\}]$$
$$s[\delta']<\delta>$$
$$= [x \notin FV(s[\delta'])]$$
$$s[\delta']<\delta'>$$
$$= s<\delta'>.$$

Thus we have $I(\mu x[s]<\delta>)=I(s<\delta'>)$. Combining this with $I'(\mu x[s])(\delta)=I'(s)(\delta')$, which we saw above, yields:

$$\Xi(\mu x[s],\delta) \Leftrightarrow \Xi(s,\delta').$$

Because $\{s\}\times\Delta\subseteq\Xi$ we may conclude: $\Xi(\mu x[s],\delta)$.

□

We formulate the relation of $\mathcal{O}_0$ and $\mathcal{O}_0'$ in terms of their defining contractions Φ_0 and Φ_0'. This can be elegantly done using the following

DEFINITION 1.29
We define $<>:(L_0^{cl}\to P_0)\to(L_0\to\Delta\to P_0)$, for every $F\in L_0^{cl}\to P_0$, by

$$<>(F) = F^{<>} \text{ (notation)}$$
$$= \lambda s\in L_0.\lambda\delta\in\Delta.\ F(s<\delta>).$$

REMARK
This mapping links two kinds of semantic functions, one using syntactic environments, and the other one not using environments. If $F\in L_0^{cl}\to P_0$, then $F^{<>}$ is a in a sense *extended* version of F: it can take as an argument also statements $s\in L_0$ that are *not* closed, provided it is supplied with a syntactic environment, which is to give the (syntactic) values for the free variables in s.

THEOREM 1.30 *(Relating Φ_0 and Φ_0')*: $\forall F\in L_0^{cl}\to P_0\ [\Phi_0'(F^{<>}) = (\Phi_0(F))^{<>}]$

PROOF
The theorem is an immediate consequence of theorem 1.28. Let $F\in L_0^{cl}\to P_0$, let $s\in L_0$, $s\neq E$.

$$\Phi_0'(F^{<>})(s)(\delta) = \bigcup\{a\cdot F^{<>}(s')(\delta')\mid(a,s',\delta')\in I'(s)(\delta)\}$$
$$= \bigcup\{a\cdot F(s'<\delta'>)\mid(a,s',\delta')\in I'(s)(\delta)\}$$
$$= [\text{theorem 1.28}]$$
$$\bigcup\{a\cdot F(s'<\delta'>)\mid(a,s'<\delta'>)\in I(s<\delta>)\}$$
$$= \Phi_0(F)(s<\delta>)$$
$$= (\Phi_0(F))^{<>}(s)(\delta).$$

Because Φ_0 and Φ_0' are contractions with $\mathcal{O}_0$ and $\mathcal{O}_0'$ as their respective fixed points, we have:

COROLLARY 1.31 ($\mathcal{O}_0'=\mathcal{O}_0^{<>}$): $\forall s\in L_0\ \forall\delta\in\Delta\ [\mathcal{O}_0'[\![s]\!](\delta) = \mathcal{O}_0[\![s<\delta>]\!]]$.

Finally we relate

$\mathcal{O}_0'$:$L_0 \to \Delta \to P_0$ and $\mathcal{D}_0$:$L_0 \to \Gamma \to P_0$.

For this purpose we define the following mapping.

DEFINITION 1.32
We define $\sim$:$(L_0 \to \Gamma \to P_0) \to (L_0 \to \Delta \to P_0)$ by:

$$\sim(F) = \tilde{F} \text{ (notation)}$$
$$= \lambda s \in L_0 \cdot \lambda \delta \in \Delta \cdot F(s)(\tilde{\delta}^F)$$

for $F \in L_0 \to \Gamma \to P_0$, where $\tilde{\delta}^F$ is given by $\tilde{\delta}^F = \lambda x \in dom(\delta) \cdot F(\delta(x))(\tilde{\delta}^F)$. (We often write $\tilde{\delta}$ rather than $\tilde{\delta}^F$ if from the context it is clear which F should be taken.)

REMARKS
(1) We have to justify the self-referential definition of $\tilde{\delta}^F$. For this purpose we define

$$\Xi(s,\delta) \equiv \forall x \in FV(s)\ [s \notin L_0^x \to (\tilde{\delta}^F(x) \text{ is well defined})],$$

for $s \in L_0$ and $\delta \in \Delta$, and use the induction principle to prove: $\Xi = L_0 \times \Delta$. Then it follows for all $x \in Stmv$ with $x \in dom(\delta)$ that $\tilde{\delta}^F(x)$ is well defined. Conditions (1) through (3) of the induction principle are trivially fulfilled. We prove condition (4). Suppose $(\delta(x),\delta) \in \Xi$. Thus $\tilde{\delta}^F(y)$ is well defined for all $y \in FV(\delta(x))$. This implies that $\tilde{\delta}^F(x)$ is well defined, since

$$\tilde{\delta}^F(x) = F(\delta(x))(\tilde{\delta}^F).$$

(2) In the same way as $<>$, also $\sim$ links two different kinds of semantic functions, one using *syntactic*, and the other using *semantic* environments. Again $\tilde{F}$ is an extended version of F in the sense that it takes syntactic environments as an argument instead of semantic ones. In the definition above a syntactic environment $\delta \in \Delta$ is changed into a *semantic version* (according to the semantic function F) $\tilde{\delta}^F$ of it, which then is supplied as an argument to F.

Next, we come to the main theorem of this chapter. It relates the denotational semantics $\mathcal{D}_0$ and the operational semantics $\mathcal{O}'_0$, which is a fixed point of Φ'_0, by stating that also $\tilde{\mathcal{D}}_0$ is a fixed point of Φ'_0. From this it follows that $\mathcal{O}_0' = \tilde{\mathcal{D}}_0$.

THEOREM 1.33: $\Phi_0'(\tilde{\mathcal{D}}_0) = \tilde{\mathcal{D}}_0$

PROOF
Let $\Xi \subseteq L_0 \times \Delta$ be defined by

$$\Xi(s,\delta) \equiv \Phi_0'(\tilde{\mathcal{D}}_0)(s)(\delta) = \tilde{\mathcal{D}}_0(s)(\delta)$$

for $(s,\delta) \in L_0 \times \Delta$. We use the induction principle for $L_0 \times \Delta$ to show that $\Xi = L_0 \times \Delta$. Let $\delta \in \Delta$.
(1) For $a \in A$ we have $\Phi_0'(\tilde{\mathcal{D}}_0)(a)(\delta) = \{a\} = \tilde{\mathcal{D}}_0(a)(\delta)$, so $A \times \Delta \subseteq \Xi$.
(2) Let $s, \bar{s} \in L_0$ and suppose $\Xi(s,\delta)$. We show: $\Xi(s;\bar{s},\delta)$.

$$\Phi_0'(\tilde{\mathcal{D}}_0)(s;\bar{s})(\delta) = \text{[definition } \Phi_0' \text{ and } I'(s;\bar{s})\text{]}$$
$$\bigcup \{a' \cdot \tilde{\mathcal{D}}_0(s';\bar{s})(\delta') \mid (a',s',\delta') \in I'(s)(\delta)\}$$
$$= \bigcup \{a' \cdot (\tilde{\mathcal{D}}_0(s')(\delta');\ \tilde{\mathcal{D}}_0(\bar{s})(\delta')) \mid (a',s',\delta') \in I'(s)(\delta)\}$$
$$= \text{[see remark (3) after definition 1.25]}$$
$$\bigcup \{a' \cdot (\tilde{\mathcal{D}}_0(s')(\delta');\ \tilde{\mathcal{D}}_0(\bar{s})(\delta)) \mid (a',s',\delta') \in I'(s)(\delta)\}$$
$$= \text{[definition ;]}$$

$$
\begin{aligned}
&(\bigcup\{a'\cdot\tilde{\mathfrak{D}}_0(s')(\delta') \mid (a',s',\delta')\in I'(s)(\delta)\});\ \tilde{\mathfrak{D}}_0(\overline{s})(\delta)\\
&= \text{[definition } \Phi_0'\text{]}\\
&\quad \Phi_0'(\tilde{\mathfrak{D}}_0)(s)(\delta);\ \tilde{\mathfrak{D}}_0(\overline{s})(\delta)\\
&= \text{[because } \Xi(s,\delta)\text{]}\\
&\quad \tilde{\mathfrak{D}}_0(s)(\delta);\ \tilde{\mathfrak{D}}_0(\overline{s})(\delta)\\
&= \tilde{\mathfrak{D}}_0(s;\overline{s})(\delta).
\end{aligned}
$$

This proves $\Xi(s;\overline{s},\delta)$. Now let $s,t\in L_0$ and suppose $\Xi(s,\delta)$ and $\Xi(t,\delta)$. We show: $\Xi(s\|t,\delta)$.

$$
\begin{aligned}
\Phi_0'(\tilde{\mathfrak{D}}_0)(s\|t)(\delta) &= \text{[definition } \Phi_0' \text{ and } I'(s\|t)\text{]}\\
&\quad \bigcup\{a'\cdot\tilde{\mathfrak{D}}_0(s'\|t)(\delta') \mid (a',s',\delta')\in I'(s)(\delta)\}\cup\\
&\quad \bigcup\{b'\cdot\tilde{\mathfrak{D}}_0(s\|t')(\delta') \mid (b',t',\delta')\in I'(t)(\delta)\}\\
&= \bigcup\{a'\cdot(\tilde{\mathfrak{D}}_0(s')(\delta')\|\tilde{\mathfrak{D}}_0(t)(\delta')) \mid (a',s',\delta')\in I'(s)(\delta)\}\cup\\
&\quad \bigcup\{b'\cdot(\tilde{\mathfrak{D}}_0(s)(\delta')\|\tilde{\mathfrak{D}}_0(t')(\delta')) \mid (b',t',\delta')\in I'(t)(\delta)\}\\
&= \text{[see remark (3) after definition 1.25]}\\
&\quad \bigcup\{a'\cdot(\tilde{\mathfrak{D}}_0(s')(\delta')\|\tilde{\mathfrak{D}}_0(t)(\delta)) \mid (a',s',\delta')\in I'(s)(\delta)\}\cup\\
&\quad \bigcup\{b'\cdot(\tilde{\mathfrak{D}}_0(s)(\delta)\|\tilde{\mathfrak{D}}_0(t')(\delta')) \mid (b',t',\delta')\in I'(t)(\delta)\}\\
&= \text{[definition } \mathbb{L} \text{ (see remark 1.17(2))]}\\
&\quad ((\bigcup\{a'\cdot\tilde{\mathfrak{D}}_0(s')(\delta') \mid (a',s',\delta')\in I'(s)(\delta)\})\mathbb{L}\ \tilde{\mathfrak{D}}_0(t)(\delta))\ \cup\\
&\quad ((\bigcup\{b'\cdot\tilde{\mathfrak{D}}_0(t')(\delta') \mid (b',t',\delta')\in I'(t)(\delta)\})\mathbb{L}\ \tilde{\mathfrak{D}}_0(s)(\delta))\\
&= \text{[definition } \Phi_0'\text{]}\\
&\quad (\Phi_0'(\tilde{\mathfrak{D}}_0)(s)(\delta)\mathbb{L}\ \tilde{\mathfrak{D}}_0(t)(\delta))\ \cup\\
&\quad (\Phi_0'(\tilde{\mathfrak{D}}_0)(t)(\delta)\mathbb{L}\ \tilde{\mathfrak{D}}_0(s)(\delta))\\
&= \text{[we have } \Xi(s,\delta) \text{ and } \Xi(t,\delta)\text{]}\\
&\quad (\tilde{\mathfrak{D}}_0(s)(\delta)\mathbb{L}\ \tilde{\mathfrak{D}}_0(t)(\delta))\ \cup\\
&\quad (\tilde{\mathfrak{D}}_0(t)(\delta)\mathbb{L}\ \tilde{\mathfrak{D}}_0(s)(\delta))\\
&= \tilde{\mathfrak{D}}_0(s)(\delta)\|\tilde{\mathfrak{D}}_0(t)(\delta)\\
&= \tilde{\mathfrak{D}}_0(s\|t)(\delta).
\end{aligned}
$$

This proves $\Xi(s\|t,\delta)$. The case $\Xi(s\cup t,\delta)$ is simple.

(3) Let $s\in L_0^{\mathfrak{X}}$ and suppose $\{s\}\times\Delta\subseteq\Xi$. We show: $\Xi(\mu x[s],\delta)$. Assume (without loss of generality) that $x\notin dom(\delta)$. Then

$$
\begin{aligned}
\Phi_0'(\tilde{\mathfrak{D}}_0)(\mu x[s])(\delta) &= \text{[definition } \Phi_0' \text{ and } I'(\mu x[s])(\delta)\text{; let } \delta'=\delta\{\mu x[s]/x\}\text{]}\\
&\quad \bigcup\{a'\cdot\tilde{\mathfrak{D}}_0(s')(\delta') \mid (a',s',\delta')\in I'(s)(\delta')\}\\
&= \Phi_0'(\tilde{\mathfrak{D}}_0)(s)(\delta')\\
&= \text{[we have } \Xi(s,\delta')\text{]}\\
&\quad \tilde{\mathfrak{D}}_0(s)(\delta')
\end{aligned}
$$

$$
\begin{aligned}
&= \mathcal{D}_0[\![s]\!](\tilde{\delta}') \\
&= [\text{definition } \tilde{\delta}'] \\
&\quad \mathcal{D}_0[\![s]\!](\tilde{\delta}\{\mathcal{D}_0[\![\mu x[s]]\!](\tilde{\delta})/x\}) \\
&= [\text{definition } \mathcal{D}_0] \\
&\quad \mathcal{D}_0[\![\mu x[s]]\!](\tilde{\delta}) \\
&= \tilde{\mathcal{D}}_0(\mu x[s])(\delta)
\end{aligned}
$$

This proves $\Xi(\mu x[s],\delta)$.

(4) Let $x \in Stmv$, suppose $\Xi(\delta(x),\delta)$. Now

$$
\begin{aligned}
\Phi_0'(\tilde{\mathcal{D}}_0)(x)(\delta) &= [\text{definition } \Phi_0' \text{ and } I'(x)(\delta)] \\
&\quad \Phi_0'(\tilde{\mathcal{D}}_0)(\delta(x))(\delta) \\
&= [\text{because } \Xi(\delta(x),\delta)] \\
&\quad \tilde{\mathcal{D}}_0(\delta(x))(\delta) \\
&= \mathcal{D}_0[\![\delta(x)]\!](\tilde{\delta}) \\
&= [\text{definition } \tilde{\delta}] \\
&\quad \tilde{\delta}(x) \\
&= \mathcal{D}_0[\![x]\!](\tilde{\delta}) \\
&= \tilde{\mathcal{D}}_0(x)(\delta).
\end{aligned}
$$

Thus $\Xi(x,\delta)$.

The induction principle now implies: $\Xi = L_0 \times \Delta$.

□

As an immediate consequence of this theorem, we have

COROLLARY 1.34 ($\mathcal{O}_0' = \tilde{\mathcal{D}}_0$): $\forall s \in L_0\ \forall \delta \in \Delta\ [\mathcal{O}_0'[\![s]\!](\delta) = \mathcal{D}_0[\![s]\!](\tilde{\delta})]$.

Now combining corollaries 1.31 and 1.34 yields the main theorem of this section.

THEOREM 1.35 ($\mathcal{O}_0^{<>} = \tilde{\mathcal{D}}_0$): $\forall s \in L_0\ \forall \delta \in \Delta\ [\mathcal{O}_0[\![s<\delta>]\!] = \mathcal{D}_0[\![s]\!](\tilde{\delta})]$.

COROLLARY 1.36: *For all* $s \in L_0^{\prime}$, *and arbitrary* $\gamma \in \Gamma$: $\mathcal{O}_0[\![s]\!] = \mathcal{D}_0[\![s]\!](\gamma)$.

1.5 *Summary of section* 1

It may be useful to give a short overview of this section because we shall follow the same approach of proving semantic equivalence in the next sections. We have defined an operational semantics $\mathcal{O}_0$ for L_0 as the fixed point of Φ_0, and a denotational semantics $\mathcal{D}_0$ as the fixed point of Ψ_0. We have related $\mathcal{O}_0$ and $\mathcal{D}_0$ via an intermediate semantic function $\mathcal{O}_0'$, defined as the fixed point of Φ_0'. To be more precise, we have related Φ_0, Ψ_0, and Φ_0' using mappings $<>$ and $\sim$, for which we have proved some properties, schematically represented by the following diagram:

$$\begin{array}{ccc} L_0^{cl}\to P_0 & \overset{\Phi_0}{\to} & L_0^{cl}\to P_0 \\ <>\downarrow & * & \downarrow<> \\ L_0\to\Delta\to P_0 & \overset{\Phi_0'}{\to} & L_0\to\Delta\to P_0 \\ \sim\uparrow & *_{fix} & \uparrow\sim \\ L_0\to\Gamma\to P_0 & \overset{\Psi_0}{\to} & L_0\to\Gamma\to P_0 \end{array}$$

The * in the upper rectangle indicates that it commutes, the symbol $*_{fix}$ in the lower rectangle indicates that it commutes only for the fixed point of Ψ_0 (that is, $\mathcal{D}_0$). Please note that * has been formulated as theorem 1.30, and $*_{fix}$ as theorem 1.33. The main result of section 1 (theorem 1.35) follows from this diagram, because * implies: $\Theta_0^{<>} = \Theta_0'$ and $*_{fix}$ implies: $\Theta_0' = \tilde{\mathcal{D}}_0$.

REMARK
The lower rectangle does *not* commute for arbitrary $F\in L_0\to\Gamma\to P_0$. As an example take $F=\lambda s\cdot\lambda\gamma\cdot\{\epsilon\}$. Then, for given $a,b\in A$ and $\delta\in\Delta$:

$$\begin{aligned}\tilde{\Psi_0}(F)(a;b)(\delta) &= \Psi_0(F)(a;b)(\tilde{\delta}^{\Psi_0(F)}) \\ &= \Psi_0(F)(a)(\tilde{\delta}^{\Psi_0(F)})\,\tilde{;}\,\Psi_0(F)(b)(\tilde{\delta}^{\Psi_0(F)}) \\ &= \{a\}\tilde{;}\{b\} \\ &= \{ab\},\end{aligned}$$

whereas

$$\begin{aligned}\Phi_0'(\tilde{F})(a;b)(\delta) &= \{a\cdot\tilde{F}(b)(\delta)\} \\ &= \{a\cdot F(b)(\tilde{\delta}^F)\} \\ &= \{a\}.\end{aligned}$$

2. A LANGUAGE WITH COMMUNICATION AND GLOBAL NONDETERMINISM (L_1)

2.1 *Syntax*

For L_1 we introduce some structure to the (possibly infinite) alphabet A of elementary actions. Let $C\subseteq A$ be a subset of so-called *communications*. From now on let c range over C and a,b over A. Similarly to CCS [Mi] or CSP [Ho] we stipulate a bijection $^-:C\to C$ with $^-\circ^- = id_C$. It yields for every $c\in C$ a *matching* communication $\bar{}(c)$, which will be denoted by $\bar{c}$. In $A\setminus C$ we have a special element τ denoting a successful communication. Let *Stmv*, with typical elements $x,y,\ldots$, be again the set of statement variables.

DEFINITION 2.1 (Syntax for L_1)
The set L_1, with typical elements $s,t,\ldots$, is given by

$$s ::= a \mid s_1;s_2 \mid s_1+s_2 \mid s_1\|s_2 \mid x \mid \mu x[t]$$

where $t\in L_1^x$, which is defined below. Please note that $a\in A\supseteq C$.

DEFINITION 2.2 (Syntax for L_1^x)
The set L_1^x of statements which are guarded for x is given by

$$t ::= a$$
$$\mid t;s, \text{ for } s \in L_1$$
$$\mid t_1 + t_2 \mid t_1 \| t_2$$
$$\mid y, \text{ for } y \neq x$$
$$\mid \mu x[t]$$
$$\mid \mu y[t'], \text{ for } y \neq x,\ t' \in L_1^x \cap L_1^y$$

DEFINITION 2.3 (Syntax for L_1^g)
The set L_1^g of statements which are guarded for all $x \in Stmv$ is defined by

$$t ::= a \mid t;s \mid t_1 + t_2 \mid t_1 \| t_2 \mid \mu x[t],$$

where $s \in L_1$.

REMARK
We extend L_1, L_1^x, and L_1^g with the empty statement E (see the remark following definition 1.2).

The definitions of $FV(s)$ (free variables of s) and of (syntactically) closed statements are as in section 1. The language L_1 differs from L_0 in two respects. First, the presence of communication actions entails a more sophisticated interpretation of $s_1 \| s_2$. Secondly, the operators of global nondeterminism $s_1 + s_2$ and of local nondeterminism $s_1 \cup s_2$ of L_0 are differently interpreted. For an extensive discussion of L_1 we refer the reader to [BKMOZ] (where, for obvious reasons, it is called L_2). After we have defined an operational semantics for L_1, we shall briefly discuss the intuitive meaning of L_1.

2.2 *Operational semantics*

DEFINITION 2.4 (Semantic universe P_1)
Let, as in definition 1.7, the set A^∞ be defined as $A^\infty = A^* \cup A^\omega$. We extend this set by allowing as the last element of a finite sequence a special element ∂, which will be used to denote *deadlock*:

$$A_\partial^\infty = A^* \cup A^* \cdot \{\partial\} \cup A^\omega.$$

Now we define a complete metric space P_1, with typical elements $p, q, \ldots$, as

$$P_1 = \mathcal{P}_{nc}(A_\partial^\infty),$$

the set of all non-empty, compact subsets of A_∂^∞. As a metric on P_1 we take $(d_{A_\partial^\infty})_H$ (see A.6(d)). We shall use P_1 as the semantic universe for the operational semantics of L_1, which will again (as for L_0) be based on a transition relation:

DEFINITION 2.5 (Transition relation for L_1^g)
We define a transition relation

$$\rightarrow \subseteq L_1^g \times A \times L_1$$

as the smallest relation satisfying

(i) $a \xrightarrow{a} E$, for $a \in A$. (Please note that it is also possible that $a \in C$!)

(ii) for all $a \in A$, $s, t \in L_1^g$ and $s', \bar{s} \in L_1$: if $s' \neq E$, then:

$$s \xrightarrow{a} s' \Rightarrow (s;\bar{s} \xrightarrow{a} s';\bar{s}$$
$$\wedge\ s + t \xrightarrow{a} s' \wedge t + s \xrightarrow{a} s')$$

$$\wedge\ s\|t \xrightarrow{a} s'\|t\ \wedge\ t\|s \xrightarrow{a} t\|s'$$

$$\wedge\ \mu x[s] \xrightarrow{a} s'[\mu x[s]/x]);$$

and if $s'=E$, then:

$$s \xrightarrow{a} E \Rightarrow (s;\overline{s} \xrightarrow{a} \overline{s}$$

$$\wedge\ s+t \xrightarrow{a} E\ \wedge\ t+s \xrightarrow{a} E$$

$$\wedge\ s\|t \xrightarrow{a} t\ \wedge\ t\|s \xrightarrow{a} t$$

$$\wedge\ \mu x[s] \xrightarrow{a} E).$$

(iii) for all $c\in C$, $s,t\in L_1^g$, $s',t'\in L_1$: if $s'\neq E\neq t'$, then:

$$(s \xrightarrow{c} s'\ \wedge\ t \xrightarrow{\overline{c}} t') \Rightarrow s\|t \xrightarrow{\tau} s'\|t',$$

and if $s'=E$, then:

$$(s \xrightarrow{c} E\ \wedge\ t \xrightarrow{\overline{c}} t') \Rightarrow s\|t \xrightarrow{\tau} t'.$$

DEFINITION 2.6 (Φ_1)
Let $\Phi_1:(L_1^{cl}\to P_1)\to(L_1^{cl}\to P_1)$ be given by

$$\Phi_1(F)(s) = \begin{cases} \{\epsilon\} & \text{if } s=E \\ \{\partial\} & \text{if } \{a \mid \exists s'[s \xrightarrow{a} s']\wedge a\notin C\}=\varnothing \\ \bigcup\{a\cdot F(s') \mid s \xrightarrow{a} s'\wedge a\notin C\} & \text{otherwise,} \end{cases}$$

for $F\in L_1^{cl}\to P_1$ and $s\in L_1^{cl}$.

DEFINITION 2.7: $\Theta_1 =$ Fixed Point(Φ_1)

EXAMPLES
The following examples illustrate the intended meaning of L_1:

$$\Theta_1[\![x]\!] = \{\partial\}$$

$$\Theta_1[\![c\|\overline{c}]\!] = \{\tau\}$$

$$\Theta_1[\![(a;c)\|(b;\overline{c})]\!] = \{ab\tau, ba\tau\}$$

$$\Theta_1[\![(a;b) + (a;c)]\!] = \{ab, a\partial\}$$

$$\Theta_1[\![a;(b+c)]\!] = \{ab\},$$

for $c\in C$, $a,b\in A\setminus C$.

Thus with global nondeterminacy + the statements $s_1=(a;b)+(a;c)$ and $s_2=a;(b+c)$ get different meanings under Θ_1. This difference can be understood as follows: If s_1 performs the elementary action a, the remaining statement is either the elementary action b or the communication c. In case of c, a deadlock occurs since no matching communication is available. However, if s_2 performs a, the remaining statement is $b+c$, which cannot deadlock because the action b is possible. Thus

communications c create deadlock only if neither a matching communication $\overline{c}$ nor an alternative elementary action b is available.
We again characterize the operational semantics by defining for each statement s a set of pairs of which the first element denotes a possible first step of s.

DEFINITION 2.8 (Initial steps)
We define a function $I: L_1^g \to \mathcal{P}_{fin}(A \times L_1)$ by induction on L_1^g.
(i) $I(E) = \emptyset$ and $I(a) = \{(a, E)\}$
(ii) Suppose $I(s) = \{(a_i, s_i)\}$, $I(t) = \{(b_j, t_j)\}$ for $s,t \in L_1^g$, $a_i, b_j \in A$, and $s_i, t_j \in L_1$. (The variables i and j range over some finite sets of indices, which we have omitted.) Then

$$I(s;\overline{s}) = \{(a_i, s_i;\overline{s})\} \text{ (for } \overline{s} \in L_1)$$
$$I(s+t) = I(s) \cup I(t)$$
$$I(s \| t) = \{(a_i, s_i \| t)\} \cup \{(b_j, s \| t_j)\} \cup \{(\tau, s_i \| t_j) \mid a_i = \overline{b}_j\}$$
$$I(\mu x[s]) = \{(a_i, s_i[\mu x[s]/x])\}.$$

LEMMA 2.9: $\forall a \in A \ \forall s \in L_1^g \ \forall s' \in L_1 \ [s \xrightarrow{a} s' \Leftrightarrow (a,s') \in I(s)]$

COROLLARY 2.10: *For* $F \in L_1^{cl} \to P_1$ *and* $s \in L_1^{cl}$, *such that* $\{a \mid \exists s'[s \xrightarrow{a} s'] \wedge a \notin C\} \neq \emptyset$, *we have:*

$$\Phi_1(F)(s) = \bigcup \{a \cdot F(s') \mid (a,s') \in I(s) \wedge a \notin C\}.$$

2.3 *Denotational semantics*

We follow [BKMOZ] in introducing a *branching* time semantics for L_1. First we have to define a suitable semantic universe. It is obtained as a solution of the following *domain equation*:

$$\overline{P} \cong \{p_0\} \cup \mathcal{P}_{co}(A \times \overline{P}). \qquad (*)$$

Such a solution we call a *domain*, and its elements are called *processes*. We can read the equation as follows: a process $p \in \overline{P}$ is either p_0, the so-called *nil* process indicating termination, or it is a (compact) set X of pairs $<a,q>$, where a is the first action taken and q is the *resumption*, describing the rest of p's actions. If X is the empty set, it indicates deadlock (as does ∂ in the operational semantics). For reasons of cardinality (*) has no solution when we take *all* subsets, rather than all *compact* subsets of $A \times \overline{P}$. Moreover, we should be more precise about the metrics involved. We should have written (*) like this:

DEFINITION 2.11 (Semantic universe $\overline{P}_1$)
Let $(\overline{P}_1, d)$ be a complete metric space satisfying the following reflexive domain equation:

$$\overline{P} \cong \{p_0\} \,\overline{\cup}\, \mathcal{P}_{co}(A \times id_{1/2}(\overline{P})),$$

where, for any positive real number c, id_c maps a metric space (M,d) onto (M,d') with $d'(x,y) = c \cdot d(x,y)$, and $\overline{\cup}$ denotes the *disjoint* union (see definition A.6). (For a formal definition of the metric on $\overline{P}$ we refer the reader to the appendix.) Typical elements of $\overline{P}_1$ are p and q, and are called *processes*.

We shall not go into the details of solving this equation. In [BZ] it was first described how to solve this type of equations in a metric setting. In [AR] this approach is reformulated and extended in a category-theoretic setting.

As in definition 1.16 we define a number of operators on $\overline{P}_1$.

DEFINITION 2.12 (Semantic operators)
The operators $\tilde{;}$, $\tilde{+}$, $\tilde{\|}$: $\overline{P}_1 \times \overline{P}_1 \to \overline{P}_1$ are defined as follows. Let $p,q \in \overline{P}_1$, then:

$$(i) \quad p\tilde{;}q = \begin{cases} q & \text{if } p = p_0 \\ \{<a, p'\tilde{;} q> \mid <a,p'> \in p\} & \text{otherwise} \end{cases}$$

$$(ii) \quad p\tilde{+}q = \begin{cases} p & \text{if } q = p_0 \\ q & \text{if } p = p_0 \\ p \cup q & \text{otherwise} \end{cases}$$

$$(iii) \quad p\tilde{\|}q = \begin{cases} p & \text{if } q = p_0 \\ q & \text{if } p = p_0 \\ \{<a, p'\tilde{\|}q \mid <a,p'> \in p\} \cup & \\ \{<a, p\tilde{\|}q' \mid <a,q'> \in q\} \cup & \\ \{<\tau, p'\tilde{\|}q'> \mid <c,p'> \in p \wedge <\bar{c},q'> \in q\} & \text{otherwise.} \end{cases}$$

(We often write op rather than $\tilde{op}$ if no confusion is possible.) For a justification of these definitions see remark 1.17.

DEFINITION 2.13 (Semantic environments)
We use Γ to denote the set of semantic environments (as in definition 1.19), with typical elements γ, given by

$$\Gamma = Stmv \to^{fin} \overline{P}_1.$$

DEFINITION 2.14 ($\Psi_1, \mathcal{D}_1$)
We define the denotational semantics $\mathcal{D}_1$ of L_1 as

$$\mathcal{D}_1 = \text{Fixed Point}(\Psi_1),$$

where $\Psi_1 : L_1 \to \Gamma \to \overline{P}_1$ is defined exactly as Ψ_0 in definition 1.20 but for the following two clauses:

$$\Psi_1(F)(a)(\gamma) = \{<a,p_0>\}$$

$$\Psi_1(F)(E)(\gamma) = p_0.$$

We realize that it must be difficult for the reader who sees this type of denotational semantics for the first time to understand and appreciate it. Nevertheless, we consider it for our purposes preferable to refer the reader to [BKMOZ], where he can find an extensive explanation. In this paper, we want to stress the technique of proving semantic equivalences, with which we now proceed.

2.4 *Semantic equivalence of* $\mathcal{O}_1$ and $\mathcal{D}_1$

It is quite obvious that the result of the previous section, as formulated in corollary 1.36, namely that

$$\forall s \in L_0^l \ \forall \gamma \in \Gamma \ [\mathcal{O}_0[\![s]\!] = \mathcal{D}_0[\![s]\!](\gamma)],$$

does not hold for the semantic functions $\mathcal{O}_1$ and $\mathcal{D}_1$. The semantic universe P_1 of $\mathcal{O}_1$ is a set of sets of streams, whereas $\overline{P}_1$, the semantic universe for $\mathcal{D}_1$, is a set of tree-like, branching processes. Thus, when comparing the types of $\mathcal{O}_1 : L_1 \to P_1$ and $\mathcal{D}_1 : L_1 \to \Gamma \to \overline{P}_1$, we observe that besides the fact that $\mathcal{D}_1$ takes a statement as an argument as well as an environment, which $\mathcal{O}_1$ does not (as is the case with $\mathcal{D}_0$ and $\mathcal{O}_0$), there is a second difference between $\mathcal{O}_1$ and $\mathcal{D}_1$. That is, they have different co-domains:

$P_1 \neq \overline{P}_1$ (which is *not* the case in the previous section). The strategy we shall follow to relate $\mathcal{O}_1$ and $\mathcal{D}_1$ is to define functions

$$\mathcal{O}_1' : L_1 \to \Delta \to P_1$$

(where Δ will again be a set of syntactic environments) and

$$\mathcal{D}_1' : L_1 \to \Delta \to \overline{P}_1,$$

and then relate $\mathcal{O}_1$ and $\mathcal{O}_1'$ (similarly as with $\mathcal{O}_0$ and $\mathcal{O}_0'$), next $\mathcal{D}_1'$ and $\mathcal{D}_1$ (similarly as with $\mathcal{O}_0'$ and $\mathcal{D}_0$), and finally compare $\mathcal{O}_1'$ and $\mathcal{D}_1'$ by using a suitable abstraction operator $\alpha : \overline{P}_1 \to P_1$. Like we did in the previous section we define $\mathcal{O}_1'$ (and $\mathcal{D}_1'$) as fixed point of a contraction.
We start with the comparison of $\mathcal{O}_1$ and $\mathcal{O}_1'$.

DEFINITION 2.15 (Syntactic environments)
The set Δ of syntactic environments, with typical elements δ, is given by

$$\Delta = \{\delta \mid \delta \in (Stmv \to^{\text{fin}} L_1) \wedge (\delta \text{ is normal})\}.$$

(For the notion of *normal* see definition 1.23.)

We formulate an induction principle for $L_1 \times \Delta$, as in 1.24.

THEOREM 2.16 (Induction principle for $L_1 \times \Delta$)
Let $\Xi \subseteq L_1 \times \Delta$. *If*

(1) $A \times \Delta \subseteq \Xi$

(2) $\{s,t\} \times \Delta \subseteq \Xi \Rightarrow \{s;\overline{s},\ s+t,\ s \| t\} \times \Delta \subseteq \Xi$, for $s,t,\overline{s} \in L_1$

(3) $\{s\} \times \Delta \subseteq \Xi \Rightarrow \{\mu x[s]\} \times \Delta \subseteq \Xi$, for $s \in L_0^x$

(4) $(\delta(x),\delta) \in \Xi \Rightarrow (x,\delta) \in \Xi$, for $x \in Stmv$, and $\delta \in \Delta$

then:

$$\Xi = L_1 \times \Delta.$$

PROOF: See theorem 1.24.

DEFINITION 2.17 (Initial steps with syntactic environments)
As in definition 1.25 we use the induction principle to define a function

$$I' : L_1 \to \Delta \to \mathcal{P}_{\text{fin}}(A \times L_1 \times \Delta).$$

(1) $I'(E)(\delta) = \varnothing$, and $I'(a)(\delta) = \{(a,E,\delta)\}$ for all $a \in A$, $\delta \in \Delta$.
(2) Suppose $I'(s) = \lambda\delta\cdot\{(a_i,s_i,\delta_i)\}$ and $I'(t) = \lambda\delta\cdot\{(b_j,t_j,\delta_j)\}$ for $s,t \in L_1$, $a_i,b_j \in A$, and $\delta_i,\delta_j \in \Delta$. Then:

$$I'(s;s)(\delta) = \{(a_i,\ s_i;\overline{s},\ \delta_i)\} \text{ (for all } \overline{s} \in L_1)$$

$$I'(s+t)(\delta) = I'(s)(\delta) \cup I'(t)(\delta)$$

$$I'(s \| t)(\delta) = \{(a_i,\ s_i \| t,\ \delta_i)\} \cup \{(b_j,\ s \| t_j,\ \delta_j)\} \cup \{(\tau,\ s_i \| t_j,\ \delta_i \cup \delta_j) \mid \overline{a}_i = b_j\}$$

(3), (4): as in definition 1.25.

REMARK
In the clause for $s \| t$ in the above definition we take the union of two environments, δ_i and δ_j. This we can always do, if we impose the restriction upon all δ_i's and δ_j's that:

$$\text{if } \overline{a}_i = b_j, \text{ then } (dom(\delta_i) \setminus dom(\delta)) \cap (dom(\delta_j) \setminus dom(\delta)) = \varnothing.$$

If this condition is not satisfied (and in general it is not) a suitable renaming of variables should be applied. An example of a statement for which this should happen is: $\mu x[c;x]\|\mu x[\overline{c};x]$.

DEFINITION 2.18 (Φ_1')
We define $\Phi_1':(L_1\to\Delta\to P_1)\to(L_1\to\Delta\to P_1)$ by

$$\Phi_1'(F)(s)(\delta) = \begin{cases} \{\epsilon\} & \text{if } s = E \\ \{\partial\} & \text{if } \{(a,s',\delta')\in I'(s)(\delta)\mid a\notin C\} = \varnothing \\ \bigcup\{a\cdot F(s')(\delta')\mid & (a,s',\delta')\in I'(s)(\delta)\wedge a\notin C\} \text{ otherwise} \end{cases}$$

for $F\in L_1\to\Delta\to P_1$, $s\in L_1$, and $\delta\in\Delta$.

DEFINITION 2.19: $\mathcal{O}_1'=$ Fixed Point(Φ_1')

THEOREM 2.20 *(Relating I and I')*

$$\forall s\in L_1\ \forall\delta\in\Delta\ [I'(s)(\delta) = \{(a_i,s_i,\delta_i)\} \Leftrightarrow I(s<\delta>) = \{(a_i,s_i<\delta_i>)\}]$$

PROOF: See theorem 1.28.

DEFINITION 2.21: We define $<>:(L_1^{cl}\to P_1)\to(L_1\to\Delta\to P_1)$ by

$$<>F = F^{<>}$$
$$= \lambda s\in L_1\cdot\lambda\delta\in\Delta\cdot F(s<\delta>)$$

for $F\in L_1^{cl}\to P_1$.

THEOREM 2.22 (Relating Φ_1 and Φ_1'): $\forall F\in L_1^{cl}\to P_1\ [\Phi_1'(F^{<>}) = (\Phi_1(F))^{<>}]$

PROOF: See theorem 1.30.

COROLLARY 2.23 ($\mathcal{O}_1'=\mathcal{O}_1^{<>}$): $\forall s\in L_1\ \forall\delta\in\Delta\ [\mathcal{O}_1'[\![s]\!](\delta) = \mathcal{O}_1[\![s<\delta>]\!]]$

Next we define $\mathcal{D}_1':L_1\to\Delta\to\overline{P}_1$ as the fixed point of the contraction below and compare $\mathcal{D}_1$ and $\mathcal{D}_1'$.

DEFINITION 2.24 (Ψ_1')
We define $\Psi_1':(L_1\to\Delta\to\overline{P}_1)\to(L_1\to\Delta\to\overline{P}_1)$ by

$$\Psi_1'(F)(s)(\delta) = \begin{cases} \{\epsilon\} & \text{if } s = E \\ \{<a,F(s')(\delta')>\mid(a,s',\delta')\in I'(s)(\delta)\} & \text{otherwise,} \end{cases}$$

for $F\in L_1\to\Delta\to\overline{P}_1$, $s\in L_1$, and $\delta\in\Delta$.

DEFINITION 2.25: $\mathcal{D}_1'=$ Fixed Point(Ψ_1')

REMARK
As $\mathcal{O}_1'$ also $\mathcal{D}_1'$ takes *syntactic* environments as arguments. Their co-domains, however, are different: $P_1\neq\overline{P}_1$. One could call $\mathcal{D}_1'$ a branching variant of $\mathcal{O}_1'$. Another difference is that $\mathcal{O}_1'(c)(\delta)=\{\partial\}$, whereas $\mathcal{D}_1'(c)(\delta)=\{<c,p_0>\}$, for $c\in C$ and $\delta\in\Delta$.

In order to relate $\mathcal{D}_1':L_1\to\Delta\to\overline{P}_1$ and $\mathcal{D}_1:L_1\to\Gamma\to\overline{P}_1$ we use the following

DEFINITION 2.26
Let $\sim:(L_1\to\Gamma\to\overline{P}_1)\to(L_1\to\Delta\to\overline{P}_1)$ be given by

$$\sim(F) = \tilde{F}$$
$$= \lambda s\in L_1\cdot\lambda\delta\in\Delta\cdot F(s)(\tilde{\delta}^F)$$

for $F\in L_1\to\Gamma\to\overline{P}_1$, where $\tilde{\delta}^F$ is defined as $\tilde{\delta}^F = \lambda x\in dom(\delta)\cdot F(\delta(x))(\tilde{\delta}^F)$. (For a justification of the definition of $\tilde{\delta}^F$ see remark (1) following definition 1.31.)

THEOREM 2.27: $\Phi_1'(\tilde{\mathfrak{P}}_1) = \tilde{\mathfrak{P}}_1$

PROOF: This theorem can be proved in essentially the same way as theorem 1.33.

COROLLARY 2.28: $\mathfrak{P}_1' = \tilde{\mathfrak{P}}_1$

Finally we provide the only missing link in the chain that is to connect $\mathfrak{O}_1$ with $\mathfrak{P}_1$: the comparison of

$$\mathfrak{O}_1':L_1\to\Delta\to P_1 \text{ and } \mathfrak{P}_1':L_1\to\Delta\to\overline{P}_1.$$

We relate their different semantic universes P_1 and $\overline{P}_1$ in the following

DEFINITION 2.29 (Abstraction operator α)
We define an abstraction operator $\alpha:\overline{P}_1\to P_1$ by:

$$\alpha = streams\circ restr,$$

where *restr* (for *restriction*) and *streams* are recursively defined:

(*i*) $restr:\overline{P}_1\to\overline{P}_1$

$$p\mapsto\begin{cases}p_0 & \text{if } p = p_0\\ \{<a,restr(p')> \mid <a,p'>\in p\wedge a\notin C\} & \text{otherwise}\end{cases}$$

(*ii*) $streams:\overline{P}_1\to P_1$

$$p\mapsto\begin{cases}\{\epsilon\} & \text{if } p = p_0\\ \{\partial\} & \text{if } p = \varnothing\\ \bigcup\{a\cdot streams(p') \mid <a,p'>\in p\} & \text{otherwise.}\end{cases}$$

REMARKS
(1) Since the definition of *restr* and *streams* is recursive, we have to verify that it is well formed. It suffices to note that these functions can be defined as fixed points of contracting functions (cf. remark 1.17).
(2) The abstraction operator α transforms a (branching) process $p\in\overline{P}_1$ into an element $\alpha(p)\in P_1$ in two steps. First it cuts off all branches (all subprocesses) of p_1 that are labeled with an element of C: these c's can be regarded as failed (individual) attempts at communication. This is what *restr* does. Then *streams* takes all paths (streams) of the result of *restr* (p), putting a ∂ symbol (denoting deadlock) at the end of all paths ending in the empty process. This can be understood as follows: When a path in *restr*(p) ends in the empty process this means that the operation *restr* has cut off everything at the end of the corresponding path in p. By definition of *restr* only c's could have been present. Thus this path in p should be interpreted as indicating a situation in which only individual communication steps can be taken. Operationally, we consider this to be a case of deadlock. Therefore, we replace this empty process by ∂. This is what *streams* does.

Now that we have defined a mapping $\alpha:\overline{P}_1 \to P_1$, we extend it in the following way.

DEFINITION 2.30
Let $\alpha:(L_1 \to \Delta \to \overline{P}_1) \to (L_1 \to \Delta \to P_1)$ be defined by

$$\begin{aligned} \alpha(F) &= F^\alpha \text{ (notation)} \\ &= \lambda s \in L_1 \cdot \lambda \delta \in \Delta \cdot \alpha(F(s)(\delta)) \end{aligned}$$

for $F \in L_1 \to \Delta \to \overline{P}_1$. (Please note that we use again the symbol α. We trust that no confusion will arise from this slight abuse of language.)

THEOREM 2.31 (Relating Ψ_1' and Φ_1'): $\forall F \in L_1 \to \Delta \to \overline{P}_1\ [\Phi_1'(F^\alpha) = (\Psi_1'(F))^\alpha]$

PROOF
Let $F \in L_1 \to \Delta \to \overline{P}_1$, let $s \in L_1$ and $\delta \in \Delta$ be such that $\{(a,s',\delta') \in I'(s)(\delta) \mid a \notin C\} \neq \varnothing$. Then:

$$\begin{aligned} \Phi_1'(F^\alpha)(s)(\delta) &= \bigcup \{a \cdot F^\alpha(s')(\delta') \mid (a,s',\delta') \in I'(s)(\delta) \wedge a \notin C\} \\ &= \bigcup \{a \cdot (\alpha(F(s')(\delta')) \mid (a,s',\delta') \in I'(s)(\delta) \wedge a \notin C\} \\ &= \mathit{streams}(\{<a, \mathit{restr}(F(s')(\delta'))> \mid (a,s',\delta') \in I'(s)(\delta) \wedge a \notin C\}) \\ &= \mathit{streams} \circ \mathit{restr}(\{<a, F(s')(\delta')> \mid (a,s,\delta') \in I'(s)(\delta)\}) \\ &= \alpha(\Psi_1'(F)(s)(\delta)) \\ &= (\Psi_1'(F))^\alpha(s)(\delta). \end{aligned}$$

If $s \in L_1$ and $\delta \in \Delta$ are such that $\{(a,s',\delta') \in I'(s)(\delta) \mid a \notin C\} = \varnothing$, then

$$\begin{aligned} \Phi_1'(F)(s)(\delta) &= \{\partial\} \\ &= \mathit{streams}(\varnothing) \\ &= \mathit{streams} \circ \mathit{restr}(\{<a, F(s')(\delta')> \mid (a,s',\delta') \in I'(s)(\delta)\}) \\ &= (\Psi_1'(F))^\alpha(s)(\delta). \end{aligned}$$

COROLLARY 2.32 $((\mathcal{D}_1')^\alpha = \Theta_1')$: $\forall s \in L_1\ \forall \delta \in \Delta\ [\alpha(\mathcal{D}_1'[\![s]\!](\delta) = \Theta_1'[\![s]\!](\delta)]$

Combining corollaries 2.23, 2.28 and 2.32, which state:

(2.23) $\Theta_1^{<>} = \Theta_1'$

(2.32) $\Theta_1' = (\mathcal{D}_1')^\alpha$

(2.28) $\mathcal{D}_1' = \tilde{\mathcal{D}}_1$,

now yields the main theorem of this section:

THEOREM 2.33 $(\Theta_1^{<>} = (\tilde{\mathcal{D}}_1)^\alpha)$: $\forall s \in L_1\ \forall \delta \in \Delta\ [\Theta_1[\![s<\delta>]\!] = \alpha(\mathcal{D}_1[\![s]\!](\tilde{\delta}))]$

COROLLARY 2.34: For all $s \in L_1^{cl}$ and arbitrary $\gamma \in \Gamma$: $\Theta_1[\![s]\!] = \alpha(\mathcal{D}_1[\![s]\!](\gamma))$.

2.5 *Summary of section* 2

We can again give a quick overview of the main theorems of this section by drawing a diagram as follows:

$$
\begin{array}{lcll}
 & \overset{\Phi_1}{\rightarrow} & & \\
L_1^{cl}\rightarrow P_1 & \rightarrow & L_1^{cl}\rightarrow P_1 & \\
<>\downarrow & * & \downarrow<> & \text{(theorem 2.22)} \\
 & \overset{\Phi_1'}{\rightarrow} & & \\
L_1\rightarrow\Delta\rightarrow P_1 & \rightarrow & L_1\rightarrow\Delta\rightarrow P_1 & \\
\alpha\uparrow & * & \uparrow\alpha & \text{(theorem 2.31)} \\
 & \overset{\Psi_1'}{\rightarrow} & & \\
L_1\rightarrow\Delta\rightarrow\overline{P}_1 & \rightarrow & L_1\rightarrow\Delta\rightarrow\overline{P}_1 & \\
\sim\uparrow & *_{fix} & \uparrow\sim & \text{(theorem2.27)} \\
 & \overset{\Psi_1}{\rightarrow} & & \\
L_1\rightarrow\Gamma\rightarrow\overline{P}_1 & \rightarrow & L_1\rightarrow\Gamma\rightarrow\overline{P}_1 &
\end{array}
$$

where (as in subsection 1.5) * indicates commutativity and $*_{fix}$ indicates commutativity with respect to the fixed point of Ψ_1 (that is, $\mathfrak{D}_1$). Please note that if we could identify P_1 and $\overline{P}_1$, we could identify the second and the third horizontal lines of this diagram, leaving out the mapping α. This would yield a diagram of exactly the same shape as that of subsection 1.5. This is just a way of rephrasing what has already been said above: The only new thing about proving semantic equivalence for L_1, compared with L_0, is the presence of a difference between the semantic universes P_1 and $\overline{P}_1$ of $\mathfrak{O}_1$ and $\mathfrak{D}_1$, which made the introduction of α necessary. Theorems 2.22 and 2.27 are just (slightly) modified versions of theorems already present in section 1 (namely, theorems 1.30 and 1.33).

3. A NONUNIFORM LANGUAGE WITH VALUE PASSING (L_2)

We devote the third section of our paper to the discussion of semantic equivalence for a *nonuniform* language. Elementary actions are no longer uninterpreted but taken as either assignment or tests. Communication actions c and $\overline{c}$ are refined to actions $c?v$ and $c!e$ (with v variable and e an expression), and successful communication now involves two effects: both *synchronization* (as in the language L_1) and *value passing*: the (current) value of e is assigned to v. Thus, we have here the synchronous handshaking variety of message passing in the sense of CCS or CSP.
We shall introduce a language L_2 which embodies these features and present its operational and denotational semantics $\mathfrak{O}_2$ and $\mathfrak{D}_2$. Nonuniformity of L_2 calls for the notion of *state* in both semantic models: They now deliver sets of streams, or processes, over state transformations, not over uninterpreted actions as in the previous sections. The main goal of this section is to provide the reader with yet another example of a language to which the method for proving semantic equivalence, as developed in section 1 and 2, applies. Although L_2 will be in some sense more complex than L_1 and accordingly $\mathfrak{O}_2$ and $\mathfrak{D}_2$ more intricate than $\mathfrak{O}_1$ and $\mathfrak{D}_1$, the proof of the equivalence of operational and denotational semantics will essentially be the same. Because of this emphasis on proving semantic equivalence, we shall not give very much explanation when defining the semantics. For this we refer the reader again to [BKMOZ], which we (roughly) follow in our definition of $\mathfrak{O}_2$ and $\mathfrak{D}_2$. Nor shall we give any proofs, because all of them can be obtained by straightforwardly modifying a corresponding one from section 2.

3.1 *Syntax*

We now present the syntax of L_2. We use three new syntactic categories, viz.
- the set *Var*, with elements v,w, of *individual variables*
- the set *Exp*, with elements e, of *expressions*
- the set *Bexp*, with elements b, of *boolean expressions*.

We shall not specify a syntax for *Exp* and *Bexp*. We assume that (boolean) expressions are of an

elementary kind; in particular, they have no side effects and their evaluation always terminates. Statement variables $x,y, \ldots$ are as before, as are the communications $c \in C$. The latter now appear syntactically as part of value passing communication actions $c?v$ or $c!e$.

DEFINITION 3.1 (Syntax for L_2)

$$s ::= v:=e \mid b \mid c?v \mid c!e \mid s_1;s_2 \mid s_1 + s_2 \mid s_1 \| s_2 \mid x \mid \mu x[t]$$

where $t \in L_2^x$, defined in

DEFINITION 3.2 (Syntax for L_2^x)
The set L_2^x of statements which are guarded for x is given by

$$\begin{aligned} t ::= {} & v:=e \mid b \mid c?v \mid c!e \\ & \mid t;s, \text{ for } s \in L_2 \\ & \mid t_1 + t_2 \mid t_1 \| t_2 \\ & \mid y, \text{ for } y \neq x \\ & \mid \mu x[t] \\ & \mid \mu y[t'], \text{ for } y \neq x,\ t' \in L_2^x \cap L_2^y \end{aligned}$$

DEFINITION 3.3 (Syntax for L_2^g)
The set L_2^g of statements which are guarded for all $x \in Stmv$ is defined by

$$t ::= v:==e \mid b \mid c?v \mid c!e \mid t;s \mid t_1 + t_2 \mid t_1 \| t_2 \mid \mu x[t],$$

where $s \in L_2$.

REMARK: The sets L_2, L_2^x, and L_2^g are extended with the empty statement E (cf. the remark preceding definition 1.3).

It will be useful to unite assignments $v:=e$, tests b and communications $c?v$ and $c!e$ into one set of *basic* steps.

DEFINITION 3.4 (Basic steps)
We define the set *Bsteps* of basic steps, with typical element a, by

$$BStep = Comm \cup Bexp \cup Asg,$$

where the set *Comm* of communications is defined by

$$Comm = \{c?v \mid c \in C, v \in Var\} \cup \{c!e \mid c \in C, e \in Exp\},$$

and the set *Asg*, of assignments, is defined by

$$Asg = \{v:=e \mid v \in Var, e \in Exp\}.$$

The sets *BSteps* and *Comm* can be regarded as the nonuniform equivalents of the sets A of atomic actions and C of communications of the previous section.

3.2 *Operational semantics*

DEFINITION 3.5 (Transition relation for L_2^g)
We define $\rightarrow \subseteq L_2^g \times BStep \times L_2$ as the smallest relation satisfying

(i) $a \xrightarrow{a} E$, for all $a \in BStep$. (Please note that it is also possible that $a \in Comm$!)

(ii) for all $a \in BStep$, $s, t \in L_2^s$ and $s', \bar{s} \in L_2$: if $s' \neq E$, then:

$$s \xrightarrow{a} s' \Rightarrow (s;\bar{s} \xrightarrow{a} s';\bar{s}$$
$$\wedge\ s+t \xrightarrow{a} s' \wedge t+s \xrightarrow{a} s'$$
$$\wedge\ s\|t \xrightarrow{a} s'\|t \wedge t\|s \xrightarrow{a} t\|s'$$
$$\wedge\ \mu x[s] \xrightarrow{a} s'[\mu x[s]/x]);$$

and if $s' = E$, then:

$$s \xrightarrow{a} E \Rightarrow (s;\bar{s} \xrightarrow{a} \bar{s}$$
$$\wedge\ s+t \xrightarrow{a} E \wedge t+s \xrightarrow{a} E$$
$$\wedge\ s\|t \xrightarrow{a} t \wedge t\|s \xrightarrow{a} t$$
$$\wedge\ \mu x[s] \xrightarrow{a} E).$$

(iii) for all $s,t \in L_2^s$, $s',t' \in L_2$, and $c?v, c!e \in Comm$: if $s' \neq E \neq t'$, then:

$$(s \xrightarrow{c!e} s' \wedge t \xrightarrow{c?v} t') \Rightarrow (s\|t \xrightarrow{v:=e} s'\|t' \wedge t\|s \xrightarrow{v:=e} t'\|s'),$$

and if $s' = E$, then:

$$(s \xrightarrow{c!e} E \wedge t \xrightarrow{c?v} t') \Rightarrow (s\|t \xrightarrow{v:=e} t' \wedge t\|s \xrightarrow{v:=e} t').$$

For both operational and denotational models the notion of *state* is fundamental. Elements v,w in *Var* will have values in a set *Val*. A state is a function that maps variables to their (current) values. Accordingly, we define

DEFINITION 3.6 (States)
The set Σ of *states*, with typical element σ, is defined as

$$\Sigma = Var \to Val.$$

We shall also employ a special failure state ∂, with $\partial \notin \Sigma$, and define

$$\Sigma_\partial^\infty = \Sigma^* \cup \Sigma^* \cdot \{\partial\} \cup \Sigma^\omega.$$

Elements of Σ_∂^∞ will be denoted by finite or infinite tuples $<\sigma_1,\sigma_2, \ldots>$. The empty tuple will be denoted by ϵ. We shall write σ for $<\sigma>$. Concatenation is defined as usual.

For expressions $e \in Exp$ and $b \in BExp$ we postulate a simple semantic evaluation function, details of which we do not bother to provide. The values of e and b in state σ will be denoted simply by

$$[\![e]\!]\sigma\ (\in Val) \text{ and } [\![b]\!]\sigma\ (\in \{tt,ff\}).$$

DEFINITION 3.7 (Semantic universe P_2)
We define the semantic universe P_2 by

$$P_2 = \Sigma \to \mathcal{P}_{nc}(\Sigma_\partial^\infty),$$

where $\mathcal{P}_{nc}(\Sigma_\partial^\infty)$ is the set of all non-empty and compact subsets of Σ_∂^∞.

DEFINITION 3.8 (Φ_2)
Let $\Phi_2: (L_2^{cl} \to P_2) \to (L_2^{cl} \to P_2)$ be defined by

$$\Phi_2(F)(E) = \{\epsilon\};$$

if $\{a \mid \exists s'[s \xrightarrow{a} s'] \wedge (a \in Asg \vee (a \in BExp \wedge [\![a]\!]\sigma = tt))\} = \varnothing$, then

$$\Phi_2(F)(s) = \{\partial\};$$

otherwise

$$\Phi_2(F)(s) = \bigcup \{\sigma \cdot F(s')(\sigma) \mid s \xrightarrow{b} s' \wedge [\![b]\!]\sigma = tt\} \cup \bigcup \{\sigma_{v:=e} \cdot F(s')(\sigma_{v:=e}) \mid s \xrightarrow{v:=e} s'\},$$

for $F \in L_2^{cl} \to P_2$ and $s \in L_2$, and with

$$\sigma_{v:=e} = \sigma\{[\![e]\!]\sigma/v\}.$$

(The notation $\sigma_{v:=e}$ will also be used in the sequel.)

DEFINITION 3.9: $\mathcal{O}_2$ = Fixed Point(Φ_2)

EXAMPLES

$$\mathcal{O}_2[\![v:=0]\!] = \lambda\sigma \cdot \{<\sigma\{0/v\}>\}.$$

$$\mathcal{O}_2[\![v:=0 \| (v:=1;\ v:=v+1)]\!] = \lambda\sigma \cdot \{<\sigma\{0/v\}, \sigma\{1/v\}, \sigma\{2/v\}>, <\sigma\{1/v\}, \sigma\{0/v\}, \sigma\{1/v\}>, <\sigma\{1/v\}, \sigma\{2/v\}, \sigma\{0/v\}>\}$$

$$\mathcal{O}_2[\![v:=0;\ \mu x[v:=v+1;\ x]]\!] = \lambda\sigma \cdot \{<\sigma\{0/v\}, \sigma\{1/v\}, \sigma\{2/v\}, \ \ldots >\}$$

$$\mathcal{O}_2[\![v:=0; v<0]\!] = \lambda\sigma \cdot \{<\sigma\{0/v\}, \partial>\}$$

$$\mathcal{O}_2[\![c?v]\!] = \lambda\sigma \cdot \{<\partial>\}$$

$$\mathcal{O}_2[\![c?v \| c!3]\!] = \lambda\sigma \cdot \{<\sigma\{3/v\}>\}$$

We can again characterize the operational model using an initial step function.

DEFINITION 3.10 (Initial steps)
Let $I: L_2^g \to \mathcal{P}_{fin}(BStep \times L_2)$ be defined by
(i) $I(E) = \varnothing$, $I(a) = \{(a, E)\}$, for $a \in BStep$
(ii) Suppose $I(s) = \{(a_i, s_i)\}$, $I(t) = \{(b_j, t_j)\}$ for $s, t \in L_2^g$, $a_i, b_j \in BStep$, and $s_i, t_j \in L_2$. Then

$$I(s;\overline{s}) = \{(a_i, s_i;\overline{s})\}, \text{ for } \overline{s} \in L_2$$

$$I(s+t) = I(s) \cup I(t)$$

$$I(s \| t) = \{(a_i, s_i \| t)\} \cup \{(b_j, s \| t_j)\} \cup \{(v:=e, s_i \| t_j) \mid (a_i = c?v \wedge b_j = c!e) \vee (a_i = c!e \wedge b_j = c?v)\}$$

$$I(\mu x[s]) = \{(a_i, s_i[\mu x[s]/x])\}.$$

LEMMA 3.11: $\forall a \in BStep \forall s \in L_2^g\ \forall s \in L_2\ [s \xrightarrow{a} s' \Leftrightarrow (a, s') \in I(s)]$

COROLLARY 3.12
For $F \in L_2^{cl} \to P_2$, $s \in L_2^{cl}$ *and* $\sigma \in \Sigma$ *with* $\{(a,s') \in I(s) \mid a \in Asg \vee (a \in BExp \wedge [\![a]\!]\sigma = tt)\} \neq \emptyset$:

$$\Phi_2(F)(s)(\sigma) = \bigcup \{\sigma \cdot F(s')(\sigma) \mid (b,s') \in I(s) \wedge [\![b]\!]\sigma = tt))\} \cup \bigcup \{\sigma_{v:=e} \cdot F(s')(\sigma_{v:=e}) \mid (v:=e,\ s') \in I(s)\}.$$

3.3 *Denotational semantics*

As in section 2.3 we start with the definition of a suitable semantic universe. It will be a process domain that is obtained as a solution of the following domain equation:

$$\overline{P} \cong \{p_0\} \cup \mathcal{P}_{co}(SSteps \times \overline{P}),$$

where the set *SSteps* of *semantic steps*, with typical elements κ, is given by

$$\begin{aligned} SSteps &= (\Sigma \to \Sigma) \\ &\cup (\Sigma \to \{tt,ff\}) \\ &\cup (C \times Var) \\ &\cup (C \times (\Sigma \to Val)). \end{aligned}$$

We can read this equation as follows: a process $p \in \overline{P}$ is either p_0, the nil process, or it is a (compact) set X of semantic steps $\kappa \in SSteps$. Such a semantic step can have one out of four forms. First it can be a state transformation. These will be used to give a semantics to assignments. Then it can be a mapping from states to the set of truth values, corresponding with boolean expressions. Next, it can be a pair $<c,v>$, corresponding with an input statement $c?v$. And finally it can be a pair $<c,f>$, corresponding with an output statement $c!e$. Here, f is used to denote the value of e (that is, $[\![e]\!] \in \Sigma \to Val$).
As in section 2.3 we should be more precise about the metrics involved. We give a formal definition below and refer the reader to section 2.3 for further explanation and references.

DEFINITION 3.13 (Semantic universe $\overline{P}_2$)
Let $(\overline{P}_2,d)$ be a complete metric space such that it satisfies the following domain equation:

$$\overline{P} \cong \{p_0\} \overline{\cup} \mathcal{P}_{co}(SSteps \times id_{1/2}(\overline{P})),$$

with *SSteps* as above. Typical elements of $\overline{P}_2$ will be p and q.

DEFINITION 3.14 (Semantic operators)
The operators $\tilde{;}$, $\tilde{+}$, and $\tilde{\|}: \overline{P}_2 \times \overline{P}_2 \to \overline{P}_2$ are defined as follows. Let $p,q \in \overline{P}_2$, $\kappa \in SSteps$, $c \in C$, $v \in Var$, and $f \in \Sigma \to Val$. Then:
(i)

$$p \tilde{;} q = \begin{cases} q & \text{if } p = p_0 \\ \{<\kappa, p' \tilde{;} q> \mid <\kappa,p'> \in p\} & \text{if } p \neq p_0 \end{cases}$$

(ii)

$$p \tilde{+} q = \begin{cases} p & \text{if } q = p_0 \\ q & \text{if } p = p_0 \\ p \cup q & \text{otherwise} \end{cases}$$

(iii) If $p = p_0$, then $p \tilde{\|} q = q \tilde{\|} p = q$. If $p \neq p_0$ and $q \neq p_0$, then:

$$p\tilde{\|}q = \{<\kappa, p'\|q> \mid <\kappa,p'>\in p\}\cup$$
$$\{<\kappa, p\|q'> \mid <\kappa,q'>\in q\}\cup$$
$$\{<\lambda\sigma\cdot\sigma\{f(\sigma)/v\},p'\|q'> \mid (<<c,v>,p'>\in p\wedge<<c,f>,q'>\in q)\vee$$
$$(<<c,f>,p'>\in p\wedge<<c,v>,q'>\in q)\}.$$

For a justification of these self-referential definitions see remark 1.17.

DEFINITION 3.15 (Semantic environments): $\Gamma = Stmv \to^{fin} \overline{P}_2$ (typical elements are γ).

DEFINITION 3.16 ($\Psi_2, \mathfrak{D}_2$)
We define the denotational semantics $\mathfrak{D}_2$ of L_2 as

$$\mathfrak{D}_2 = \text{Fixed Point}(\Psi_2),$$

where $\Psi_2:(L_2\to\Gamma\to\overline{P}_2)\to(L_2\to\Gamma\to\overline{P}_2)$ is given, for $F\in L_2\to\Gamma\to\overline{P}_2$, by:

(*i*) $\Psi_2(F)(a)(\gamma)=\{<\kappa_a,p_0>\}$, and $\Psi_2(F)(E)(\gamma)=p_0$,

with

$$\kappa_a = \begin{cases} \lambda\sigma\cdot\sigma_{v:=e} & \text{if } a=v:=e \\ \lambda\sigma\cdot[\![a]\!]\sigma & \text{if } a\in BExp \\ <c,v> & \text{if } a=c?v \\ <c,\lambda\sigma\cdot[\![e]\!]\sigma> & \text{if } a=c!e. \end{cases}$$

(*ii*) $\Psi_2(F)(s\ op\ t)(\gamma) = \Psi_2(F)(s)(\gamma)\ \tilde{op}\ \Psi_2(F)(t)(\gamma)$ for $op = ;, +, \|$.

(*iii*) $\Psi_2(F)(\mu x[s])(\gamma) = \Psi_2(F)(s)(\gamma\{F(\mu x[s])(\gamma)/x\})$.

Similarly to lemma 1.21 we have that Ψ_2 is contracting.

EXAMPLES

$$\mathfrak{D}_2[\![v:=0]\!](\gamma) = \{<\lambda\sigma\cdot\sigma\{0/v\},p_0>\}$$
$$\mathfrak{D}_2[\![v:=1;v:=v+1]\!](\gamma) = \{<\lambda\sigma\cdot\sigma\{1/v\},\{<\lambda\sigma'\cdot\sigma'\{\sigma'(v)+1/v\},p_0>\}>\}$$
$$\mathfrak{D}_2[\![c?v\,\|\,c!3]\!](\gamma) = \{<<c,v>,\{<<c,\lambda\sigma\cdot 3>,p_0>\}>,$$
$$<<c,\lambda\sigma\cdot 3>,\{<<c,v>,p_0>\}>,$$
$$<\lambda\sigma\cdot\sigma\{3/v\},p_0>\}$$
$$\mathfrak{D}_2[\![v:=0;\ \mu x[v:=v+1;\ x]]\!] = \{<\lambda\sigma\cdot\sigma\{0/v\},p>\},\ \text{where } p\in\overline{P}_2 \text{ satisfies}$$
$$p = \{<\lambda\sigma\cdot\sigma\{\sigma(v)+1/v\},p>\}.$$

3.4 *Semantic equivalence of* $\mathcal{O}_2$ **and** $\mathfrak{D}_2$

The proof of the semantic equivalence of $\mathcal{O}_2$ and $\mathfrak{D}_2$ is essentially the same as in the previous section. Therefore, we only give a brief outline of how to proceed, leaving out the details of some definitions, omitting all proofs, and stressing the (small) differences. We define

$$\mathcal{O}_2' = \text{Fixed Point}(\Phi_2') \text{ and } \mathfrak{D}_2' = \text{Fixed Point}(\Psi_2')$$

with Φ_2' and Ψ_2' defined as follows. Let $\Phi_2':(L_2\to\Delta\to P_2)\to(L_2\to\Delta\to P_2)$ be given by

$$\Phi_2'(F)(E)(\delta) = \{\epsilon\};$$

if $\{(a,s',\delta')\in I'(s)(\delta) \mid a\in Asg\vee(a\in BExp\wedge[\![a]\!]\sigma=tt)\} = \varnothing$, then

$$\Phi_2'(F)(s)(\delta) = \{\partial\};$$

otherwise

$$\Phi_2'(F)(s)(\delta) = \bigcup\{\sigma\cdot F(s')(\sigma)(\delta') \mid (b,s',\delta')\in I'(s)(\delta)\wedge[\![b]\!]\sigma=tt\}\cup$$
$$\bigcup\{\sigma_{v:=e}\cdot F(s')(\delta')(\sigma_{v:=e}) \mid (v:=e,s',\delta')\in I'(s)(\delta)\},$$

for $F\in L_2\to\Delta\to P_{\underline{2}}$, $s\in L_2$ and $\delta\in\Delta$ (Δ and I' can be defined similarly to definitions 2.5 and 2.17). Let $\Psi_2':(L_2\to\Delta\to\bar{P}_2)\to(L_2\to\Delta\to\bar{P}_2)$ be defined by

$$\Psi_2'(F)(s)(\delta) = \begin{cases} p_0 & \text{if } s=E \\ \{<\kappa_a,F(s')(\delta')> \mid (a,s',\delta')\in I'(s)(\delta)\} & \text{otherwise,} \end{cases}$$

(with κ_a as in definition 3.16) for $F\in L_2\to\Delta\to\bar{P}_2$, $s\in L_2$, and $\delta\in\Delta$.

The definitions of Φ_2' and Ψ_2' are somewhat more involved than their counterparts from section 2. What is different here is that a syntactic basic step does not literally coincide with the semantic step that represents its meaning. In the previous section we had elementary actions a and c both as syntactic and semantic entities. Here we have syntactic basic steps $v:=e$, b, $c!e$, and $c?v$, all of which are semantically represented in a different way.
Similarly to the definitions 2.21 and 2.26 we can define mappings

$$<>:(L_2^{cl}\to P_2)\to(L_2\to\Delta\to P_2) \quad \text{and}$$
$$\sim:(L_2\to\Gamma\to\bar{P}_2)\to(L_2\to\Delta\to\bar{P}_2),$$

and prove

$$\mathcal{O}_2' = \mathcal{O}_2^{<>} \text{ and } \mathcal{D}_2' = \tilde{\mathcal{D}}_2.$$

Finally, we can compare $\mathcal{O}_2'$ and $\mathcal{D}_2'$ by recursively defining a suitable abstraction operator $\alpha:\bar{P}_2\to P_2$ by

$$\alpha(p_0)(\sigma)=\{\epsilon\},$$

and, for $p\neq p_0$, by

$$\alpha(p)(\sigma) = \bigcup\{f(\sigma)\cdot\alpha(p')(f(\sigma)) \mid <f,p'>\in p\wedge f\in\Sigma\to\Sigma\}\cup$$
$$\bigcup\{\sigma\cdot\alpha(p')(\sigma) \mid <f,p'>\in p\wedge(f\in\Sigma\to\{ff,tt\})\wedge f(\sigma)=tt\},$$

if $\{<f,p'> \mid <f,p'>\in p\wedge(f\in\Sigma\to\Sigma\vee(f\in\Sigma\to\{ff,tt\}\wedge f(\sigma)=tt))\}\neq\varnothing$, and by

$$\alpha(p)(\sigma) = \{\partial\}, \text{ otherwise.}$$

(For a justification of this self-referential definition see remark 1.17.) In $\alpha(p)(\sigma)$ all pairs $<\kappa,p'>\in p$ with $\kappa\in\Sigma\to\{tt,ff\}$ and $\kappa(\sigma)=ff$, or $\kappa\in C\times Var$, or $\kappa\in C\times(\Sigma\to Val)$, are neglected. This corresponds with the restriction operator of definition 2.29. A second effect of applying α is that it transforms a (branching) process $p\in\bar{P}_2$ into a function $\alpha(p)\in P_2=\Sigma\to\mathcal{P}_{nc}(A_\delta^\infty)$, which yields, when supplied with an argument σ, a set of streams (in a sense the *paths* of p). In this respect α is similar to the operator *streams* of definition 2.29. Applying α has yet another effect. If $f\in\Sigma\to\Sigma$ and $<f,p'>\in p$, then $f(\sigma)\cdot\alpha(p')(f(\sigma))\in\alpha(p)(\sigma)$: the state transformation f is applied to the current state σ, and the resulting state $f(\sigma)$ is concatenated with $\alpha(p')(f(\sigma))$, in which $f(\sigma)$, being the new state, is passed through to α applied to p', the resumption of f. In this way, the effect of different state transformations occurring subsequently in p is accumulated. A simple example may illustrate this. Consider

$$p = \mathcal{D}_2[\![v:=1;v:=v+1]\!]$$

$$= \{<\lambda\sigma\cdot\sigma_{v:=1}, \{<\lambda\sigma'\cdot\sigma'_{v:=\sigma'(v)+1}, p_0>\}>\}.$$

Then

$$\begin{aligned}\alpha(p)(\sigma) &= \{<\sigma_{v:=1}, \alpha(\{<\lambda\sigma'\cdot\sigma'_{v:=\sigma'(v)+1}, p_0>\})(\sigma_{v:=1})>\}\\ &= \{<\sigma_{v:=1}, \sigma_{v:=2}, \alpha(p_0)(\sigma_{v:=2})>\}\\ &= \{<\sigma_{v:=1}, \sigma_{v:=2}>\}.\end{aligned}$$

Next, we extend α to a mapping $\alpha:(L_2\to\Delta\to\overline{P}_2)\to(L_2\to\Delta\to P_2)$ by putting for $F\in L_2\to\Delta\to\overline{P}_2$:

$$\begin{aligned}\alpha(F) &= F^\alpha\\ &= \lambda s\cdot\lambda\delta\cdot\alpha(F(s)(\delta)).\end{aligned}$$

We shall prove that

$$\forall F\in L_2\to\Delta\to\overline{P}_2\ [\Phi_2'(F^\alpha) = (\Psi_2'(F))^\alpha].$$

Let $F\in L_2\to\Delta\to\overline{P}_2$, $s\in L_2$, $\delta\in\Delta$, and $\sigma\in\Sigma$ be such that

$$\{(a,s',\delta')\in I'(s)(\delta)\mid a\in Asg\vee(a\in BExp\wedge[\![a]\!]\sigma = tt)\}\neq\emptyset.$$

Then

$$\begin{aligned}&\Phi_2'(F^\alpha)(s)(\delta)(\sigma)\\ &= \bigcup\{\sigma\cdot F^\alpha(s')(\delta')(\sigma)\mid (b,s',\delta')\in I'(s)(\delta)\wedge[\![b]\!]\sigma=tt\}\cup\\ &\quad\bigcup\{\sigma_{v:=e}\cdot F^\alpha(s')(\delta')(\sigma_{v:=e})\mid (v:=e,s',\delta')\in I'(s)(\delta)\}\\ &= \bigcup\{\sigma\cdot(\alpha(F'(s')(\delta'))(\sigma))\mid (b,s',\delta')\in I'(s)(\delta')\wedge[\![b]\!]\sigma=tt\}\cup\\ &\quad\bigcup\{\sigma_{v:=e}\cdot(\alpha(F'(s')(\delta'))(\sigma_{v:=e}))\mid (v:=e,s',\delta')\in I'(s)(\delta')\}\\ &= \alpha(\{<\kappa_a, F'(s')(\delta')>\mid (a,s',\delta')\in I'(s)(\delta')\})(\sigma)\\ &\quad\text{[with } \kappa_a \text{ as above]}\\ &= \alpha(\Psi_2'(F)(s)(\delta))(\sigma)\\ &= (\Psi_2'(F))^\alpha(s)(\delta)(\sigma).\end{aligned}$$

The case that $\Phi_2'(F)(s)(\delta)(\sigma)=\{\partial\}$ goes similarly. This proves

$$\forall F\in L_2\to\Delta\to\overline{P}_2\ [\Phi_2'(F^\alpha) = (\Psi_2'(F))^\alpha].$$

Now it follows that

$$(\mathcal{D}_2')^\alpha = \mathcal{O}_2'.$$

Collecting the results from above, we see:

$$\mathcal{O}_2^{<>} = (\tilde{\mathcal{D}}_2)^\alpha, \text{ or}$$

$$\forall s\in L_2\ \forall\delta\in\Delta\ [\mathcal{O}_2[\![s<\delta>]\!] = \alpha(\mathcal{D}_2[\![s]\!](\tilde{\delta}))],$$

with the obvious corollary, that

$$\forall s\in L_2^{cl}\ \forall\gamma\in\Gamma\ [\mathcal{O}_2[\![s]\!] = \alpha(\mathcal{D}_2[\![s]\!](\gamma))].$$

4. Conclusions

We have developed a uniform method of comparing different semantic models for imperative concurrent programming languages. We have defined operational and denotational semantic models for such languages as fixed points of contractions on complete metric spaces, and have related them by relating their corresponding contractions. Here, we benefit from the metric structure of the underlying mathematical domains, which ensures the uniqueness of the fixed point of such contractions (Banach's theorem). It turns out that once this method has been applied to a certain (simple) language (L_0), it can be easily generalized for more complex languages (L_1 and L_2). This we consider to be the strength of this approach. Currently, we are investigating possible extensions of this method to deal with yet other languages, containing, e.g., program constructs for process creation.

Our investigations are related to the question of *full abstraction*, which at the same time is a topic for further research. If L is a language with semantics $\mathcal{O}$ and $\mathcal{D}$, then we call $\mathcal{D}$ *fully abstract with respect to* $\mathcal{O}$ if

$$\forall s \in L \forall t \in L\ [\mathcal{D}[\![s]\!] = \mathcal{D}[\![t]\!] \Leftrightarrow \forall C(\cdot)\ [\mathcal{O}[\![C(s)]\!] = \mathcal{O}[\![C(t)]\!]],$$

where $C(\cdot)$ ranges over the set of *contexts* for L, that is, the set of statements in L containing one or more holes. An example would be $s;(\cdot)$, where $(\cdot)$ denotes the hole. Given such a context $C(\cdot)$ and a statement s the statement $C(s)$ is obtained by substituting s for all the holes in $C(\cdot)$. The issue of full abstraction is mostly raised with respect to a model $\mathcal{O}$ that is *operational*, expressing a notion of observability, and a model $\mathcal{D}$ that is *compositional*. Then it follows from a relation between $\mathcal{O}$ and $\mathcal{D}$ of the form $\mathcal{O} = \alpha \circ \mathcal{D}$ that for all s and $t \in L$:

$$\mathcal{D}[\![s]\!] = \mathcal{D}[\![t]\!] \Rightarrow \forall C(\cdot)\ [\mathcal{O}[\![C(s)]\!] = \mathcal{O}[\![C(t)]\!]].$$

(This property is sometimes called: *correctness* of $\mathcal{D}$ with respect to $\mathcal{O}$.) Thus, our result of proving $\mathcal{O} = \alpha \circ \mathcal{D}$ partly solves the problem of full abstraction. The reversed arrow is still an issue for further research.

5. References

[AP] K. Apt, G. Plotkin, *Countable nondeterminism and random assignment,* Journal of the Association for Computing Machinery, Vol. 33, No. 4, October 1986, pp. 724-767.

[AR] P. America, J.J.M.M. Rutten, *Solving reflexive domain equations in a category of complete metric spaces,* Report CS-R8709, Centre for Mathematics and Computer Science, Amsterdam, February 1987. (To appear in: Proceedings of the Third Workshop on Mathematical Foundations of Programming Language Semantics, Springer-Verlag, Lecture Notes in Computer Science, 1988.)

[BKMOZ] J.W. de Bakker, J.N. Kok, J.-J. Ch. Meyer, E.-R. Olderog, J.I. Zucker, *Contrasting themes in the semantics of imperative concurrency,* in: Current Trends in Concurrency (J.W. de Bakker, W.P. de Roever, G. Rozenberg, eds.), Lecture Notes in Computer Science 224, Springer-Verlag, 1986, pp. 51-121.

[BMOZ1] J.W. de Bakker, J.-J. Ch. Meyer, E.-R. Olderog, J.I. Zucker, *Transition systems, infinitary languages and the semantics of uniform concurrency,* in: Proceedings 17th ACM STOC, Providence, R.I. (1985) 252-262.

[BMOZ2] J.W. de Bakker, J.-J. Ch. Meyer, E.-R. Olderog, J.I. Zucker, *Transition systems, metric spaces and ready sets in the semantics of uniform concurrency,* Report CS-R8601, Centre for

Mathematics and Computer Science, Amsterdam, January 1986. (To appear in: Journal of Computer and System Sciences.)

[BZ] J.W. DE BAKKER, J.I. ZUCKER, *Processes and the denotational semantics of concurrency,* Information and Control 54 (1982) 70-120.

[Du] J. DUGUNDJI, *Topology,* Allen and Bacon, Rockleigh, N.J., 1966.

[En] E. ENGELKING, *General topology,* Polish Scientific Publishers, 1977.

[FHLR] N. FRANCEZ, C.A.R. HOARE, D.J. LEHMANN, W.P. DE ROEVER, *Semantics of nondeterminism, concurrency and communication,* J. CSS 19 (1979) 290-308.

[HP] M. HENNESSY, G.D. PLOTKIN, *Full abstraction for a simple parallel programming language,* in: Proceedings 8th MFCS (J. Bečvař ed.), Lecture Notes in Computer Science 74 Springer-Verlag (1979) 108-120.

[Ho] C.A.R. HOARE, *Communicating sequential processes,* Prentice Hall International, 1985.

[Mic] E. MICHAEL, *Topologies on spaces of subsets,* in: Trans. AMS 71 (1951), pp. 152-182.

[Mil] R. MILNER, *A Calculus of communicating systems,* Lecture Notes in Computer Science 92, Springer-Verlag, 1980.

[Pl1] G.D. PLOTKIN, *A powerdomain construction,* SIAM J. Comp. 5 (1976) 452-487.

[Pl2] G.D. PLOTKIN, *A structural approach to operational semantics,* Report DAIMI FN-19, Comp. Sci. Dept., Aarhus Univ. 1981.

[Pl3] G.D. PLOTKIN, *An operational semantics for CSP,* in: Formal Description of Programming Concepts II (D. Bjørner ed.) North-Holland, Amsterdam (1983) 199-223.

[Sc] D.S. SCOTT, *Domains for denotational semantics,* Proc. 9th ICALP (M. Nielsen, E.M. Schmidt, eds.), Lecture Notes in Computer Science 140, Springer-Verlag, 1982, pp. 577-613.

6. APPENDIX: MATHEMATICAL DEFINITIONS

DEFINITION A.1 (Metric space)
A *metric space* is a pair (M,d) with M a non-empty set and d a mapping $d:M\times M\to[0,1]$ (a *metric* or *distance*) that satisfies the following properties:
(a) $\forall x,y\in M\,[d(x,y)=0 \Leftrightarrow x=y]$
(b) $\forall x,y\in M\,[d(x,y)=d(y,x)]$
(c) $\forall x,y,z\in M\,[d(x,y)\leqslant d(x,z)+d(z,y)]$.
We call (M,d) an *ultra-metric space* if the following stronger version of property (c) is satisfied:
(c') $\forall x,y,z\in M\,[d(x,y)\leqslant \max\{d(x,z),d(z,y)\}]$.
Please note that we consider only metric spaces with bounded diameter: the distance between two points never exceeds 1.

EXAMPLES A.1.1
(a) Let A be an arbitrary set. The *discrete* metric d_A on A is defined as follows. Let $x,y\in A$, then

$$d_A(x,y)=\begin{cases}0 & \text{if } x=y\\ 1 & \text{if } x\neq y.\end{cases}$$

(b) Let A be an alphabet, and let $A^\infty=A^*\cup A^\omega$ denote the set of all finite and infinite words over A. Let, for $x\in A^\infty$, $x(n)$ denote the prefix of x of length n, in case $length(x)\geqslant n$, and x otherwise. We put

$$d(x,y)=2^{-sup\{n\,|\,x(n)=y(n)\}},$$

with the convention that $2^{-\infty}=0$. Then (A^∞,d) is a metric space.

DEFINITION A.2
Let (M,d) be a metric space, let $(x_i)_i$ be a sequence in M.
(a) We say that $(x_i)_i$ is a *Cauchy sequence* whenever we have:
$\forall \epsilon>0\ \exists N\in\mathbb{N}\ \forall n,m>N\ [d(x_n,x_m)<\epsilon]$.
(b) Let $x\in M$. We say that $(x_i)_i$ *converges to* x and call x the *limit* of $(x_i)_i$ whenever we have:
$\forall \epsilon>0\ \exists N\in\mathbb{N}\ \forall n>N\ [d(x,x_n)<\epsilon]$.
Such a sequence we call *convergent.* Notation: $\lim_{i\to\infty} x_i = x$.
(c) The metric space (M,d) is called *complete* whenever each Cauchy sequence converges to an element of M.

DEFINITION A.3
Let $(M_1,d_1),(M_2,d_2)$ be metric spaces.
(a) We say that (M_1,d_1) and (M_2,d_2) are *isometric* if there exists a bijection $f:M_1\to M_2$ such that:
$\forall x,y\in M_1\ [d_2(f(x),f(y))=d_1(x,y)]$. We then write $M_1\cong M_2$. When f is not a bijection (but only an injection), we call it an *isometric embedding.*
(b) Let $f:M_1\to M_2$ be a function. We call f *continuous* whenever for each sequence $(x_i)_i$ with limit x in M_1 we have that $\lim_{i\to\infty} f(x_i)=f(x)$.
(c) Let $A\geqslant 0$. With $M_1\to^A M_2$ we denote the set of functions f from M_1 to M_2 that satisfy the following property:
$\forall x,y\in M_1\ [d_2(f(x),f(y))\leqslant A\cdot d_1(x,y)]$.
Functions f in $M_1\to^1 M_2$ we call *non-distance-increasing* (NDI), functions f in $M_1\to^\epsilon M_2$ with $0\leqslant\epsilon<1$ we call *contracting*.

PROPOSITION A.4
(a) *Let $(M_1,d_1),(M_2,d_2)$ be metric spaces. For every $A\geqslant 0$ and $f\in M_1\to^A M_2$ we have: f is continuous.*
(b) *(Banach's fixed-point theorem)*
Let (M,d) be a complete metric space and $f:M\to M$ a contracting function. Then there exists an $x\in M$ such that the following holds:
(1) *$f(x)=x$ (x is a fixed point of f),*
(2) *$\forall y\in M\ [f(y)=y\Rightarrow y=x]$ (x is unique),*
(3) *$\forall x_0\in M\ [\lim_{n\to\infty} f^{(n)}(x_0)=x]$, where $f^{(n+1)}(x_0)=f(f^{(n)}(x_0))$ and $f^{(0)}(x_0)=x_0$.*

DEFINITION A.5 (Compact subsets)
A subset X of a complete metric space (M,d) is called *compact* whenever each sequence in X has a subsequence that converges to an element of X.

DEFINITION A.6
Let $(M,d),(M_1,d_1),\ldots,(M_n,d_n)$ be metric spaces.
(a) With $M_1\to M_2$ we denote the set of all continuous functions from M_1 to M_2. We define a metric d_F on $M_1\to M_2$ as follows. For every $f_1,f_2\in M_1\to M_2$

$$d_F(f_1,f_2)=\sup_{x\in M_1}\{d_2(f_1(x),f_2(x))\}.$$

For $A\geqslant 0$ the set $M_1\to^A M_2$ is a subset of $M_1\to M_2$, and a metric on $M_1\to^A M_2$ can be obtained by taking the restriction of the corresponding d_F.
(b) With $M_1\cup\cdots\cup M_n$ we denote the *disjoint union* of $M_1,\ldots,M_n$, which can be defined as $\{1\}\times M_1\cup\cdots\cup\{n\}\times M_n$. We define a metric d_U on $M_1\cup\cdots\cup M_n$ as follows. For every $x,y\in M_1\cup\cdots\cup M_n$

$$d_U(x,y)=\begin{cases} d_j(x,y) & \text{if } x,y\in\{j\}\times M_j,\ 1\leqslant j\leqslant n\\ 1 & \text{otherwise.}\end{cases}$$

(c) We define a metric d_P on $M_1\times\cdots\times M_n$ by the following clause.

For every $(x_1, \ldots, x_n), (y_1, \ldots, y_n) \in M_1 \times \cdots \times M_n$

$$d_P((x_1, \ldots, x_n),(y_1, \ldots, y_n)) = \max_i \{d_i(x_i, y_i)\}.$$

(d) Let $\mathcal{P}_{nc}(M) =^{def} \{X | X \subseteq M \wedge X$ is compact and non-empty$\}$. We define a metric d_H on $\mathcal{P}_{nc}(M)$, called the *Hausdorff distance,* as follows. For every $X, Y \in \mathcal{P}_{nc}(M)$

$$d_H(X, Y) = \max\{\sup_{x \in X}\{d(x, Y)\}, \sup_{y \in Y}\{d(y, X)\}\},$$

where $d(x, Z) =^{def} \inf_{z \in Z}\{d(x,z)\}$ for every $Z \subseteq M$, $x \in M$.
In $\mathcal{P}_{co}(M) =^{def} \{X | X \subseteq M \wedge X$ is compact$\}$ we also have the empty set as an element. We define d_H on $\mathcal{P}_{co}(M)$ as above but extended with the following case. If $X \neq \varnothing$, then

$$d_H(\varnothing, X) = d_H(X, \varnothing) = 1.$$

(e) Let $c \in [0, \infty)$. We define: $id_c(M,d) = (M, c \cdot d)$.

PROPOSITION A.7
Let (M,d), $(M_1,d_1), \ldots, (M_n,d_n)$, d_F, d_U, d_P and d_H be as in definition A.6 and suppose that (M,d), $(M_1,d_1), \ldots, (M_n,d_n)$ are complete. We have that
(a) $(M_1 \underline{\rightarrow} M_2, d_F)$, $(M_1 \rightarrow^A M_2, d_F)$,
(b) $(M_1 \cup \cdots \cup M_n, d_U)$,
(c) $(M_1 \times \cdots \times M_n, d_P)$,
(d) $(\mathcal{P}_{nc}(M), d_H)$, and $(\mathcal{P}_{co}(M), d_H)$
are complete metric spaces. If (M,d) and (M_i,d_i) are all ultra-metric spaces these composed spaces are again ultra-metric. (Strictly spoken, for the completeness of $M_1 \rightarrow M_2$ and $M_1 \rightarrow^A M_2$ we do not need the completeness of M_1. The same holds for the ultra-metric property.)

The proofs of proposition A.7 (a), (b) and (c) are straightforward. Part (d) is more involved. It can be proved with the help of the following characterization of the completeness of the Hausdorff metric.

PROPOSITION A.8
Let $(\mathcal{P}_{co}(M), d_H)$ be as in definition A.6. Let $(X_i)_i$ be a Cauchy sequence in $\mathcal{P}_{co}(M)$. We have:

$$\lim_{i \rightarrow \infty} X_i = \{\lim_{i \rightarrow \infty} x_i | x_i \in X_i, (x_i)_i \text{ a Cauchy sequence in } M\}.$$

The proof of proposition A.8 can be found in [Mic] as a generalization of a similar result (for *closed* subsets) in [Du] and [En].