Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

J.W. de Bakker, J.J.Ch. Meyer, E.R. Olderog, J.I. Zucker

Transition systems, infinitary languages and the semantics of uniform concurrency

Department of Computer Science Report CS-R8506 March

Transition Systems, Infinitary Languages and the Semantics of Uniform Concurrency

J.W. de Bakker
Centre for Mathematics and Computer Science, Amsterdam

J. J.Ch. Meyer
Free University of Amsterdam

E. R. Olderog
Christian-Albrechts-Universität, Kiel

J.I. Zucker
State University of New York, Buffalo

Transition systems as proposed by Hennessy & Plotkin are defined for a series of three languages featuring concurrency. The first has shuffle and local nondeterminancy, the second synchronization merge and local nondeterminacy, and the third synchronization merge and global nondeterminacy. The languages are all uniform in the sense that the elementary actions are uninterpreted. Throughout, infinite behaviour is taken into account and modelled with infinitary languages in the sense of Nivat. A comparison with denotational semantics is provided. For the first two languages, a linear time model suffices; for the third language a braching time model with processes in the sense of De Bakker & Zucker is described. In the comparison an important role is played by an intermediate semantics in the style of Hoare & Olderog's specification oriented semantics. A variant on the notion of ready set is employed here. Precise statements are given relating the various semantics in terms of a number of abstraction operators.

1980 Mathematics Subject Classification: 68B10, 68C01.
1982 CR Categories: D.3.1, F.3.2, F.3.3.

Key Words & Phrases: concurrency, operational semantics, denotational semantics, transition systems, uniform languages, infinitary languages, shuffle, synchronization, local nondeterminacy, global nondeterminacy, linear time, branching time, specification-oriented semantics, ready set.

Notes:
1. The research of J.W. de Bakker is partially supported by ESPRIT project 415: Parallel Architectures and Languges;
2. This report will be published in the Proceedings 17th ACM Symposium on Theory of Computing.

Report CS-R8506
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

1. INTRODUCTION

Our paper aims at presenting a thorough study of the semantics of a number of concepts in concurrency. We concentrate on shuffle and synchronization merge, local and global nondeterminacy, and deadlocks. Somewhat more specifically, we provide a systematic analysis of these concepts by confronting, for three sample languages, semantic techniques inspired by earlier work due to Hennessy and Plotkin ([13,20]) proposing an operational approach, De Bakker et al. ([3,4,5,6]) for a denotational one, and the Oxford School ([8,18,19,21]) serving - for the purposes of our paper - an intermediate role.

Our operational semantics is based on transition systems ([14]) as employed successfully in [13,20]; applications in the analysis of proof systems were developed by Apt [1,2]. Compared with previous instances, our definitions exhibit various novel features: (i) the use of a model involving languages with finite *and infinite* words (cf. Nivat [17]); (ii) the use of full recursion (based on the copy rule) rather than just iteration; (iii) an appealingly simple treatment of synchronization; (iv) a careful distinction between local and global nondeterminacy; (v) the restriction to *uniform* concurrency.

Throughout the paper we only consider uniform statements: by this we mean an approach at the *schematic* level, leaving the elementary actions uninterpreted and avoiding the introduction of notions such as assignments or states. Many interesting issues arise at this level, and we feel that it is advantageous to keep questions which arise after interpretation for a treatment at a second level (not dealt with in our paper).

We shall study three languages in increasing order of complexity:

$\mathcal{L}_0$: shuffle (arbitrary interleaving) + local nondeterminacy (section 2)

$\mathcal{L}_1$: synchronization merge + local nondeterminacy (section 3)

$\mathcal{L}_2$: synchronization merge + global nondeterminacy (section 4)

For $\mathcal{L}_i$ with typical elements s, we shall present transition system T_i and define an induced operational semantics $\mathcal{O}_i[\![s]\!]$, i=0,1,2. We shall also define three denotational semantics $\mathcal{D}_i[\![s]\!]$ based, for i=0,1 on the "linear time" (LT) model which employs sets of sequences and, for i=2, on the "branching time" (BT) model employing *processes* (commutative trees, with sets rather than multisets of successors for any node, and with certain closure properties) of [3,4,5]. Throughout our paper we provide $\mathcal{D}_i$ *only* for $\mathcal{L}_i$ when restricted to *guarded* recursion (each recursive call has to be preceded by some elementary action); we then have an attractive metric setting with unique fixed points for contractive functions based on Banach's fixed point theorem. (Our $\mathcal{O}_i$ *do* assign meaning to the unguarded case as well.)

Our main question can now be posed: Do we have that

(1.1) $\mathcal{O}_i[\![s]\!] = \mathcal{D}_i[\![s]\!]$

We shall show that (1.1) only holds for i=0. For the more sophisticated languages $\mathcal{L}_i$, i=1,2, we cannot prove (1.1). In fact, we can even show that there exists no $\mathcal{D}_i$ satisfying (1.1), i=1,2. Rather than trying to modify $\mathcal{O}_i$ (thus spoiling its intuitive operational character) we propose to replace (1.1) by

(1.2) $\mathcal{O}_i[\![s]\!] = \alpha_i(\mathcal{D}_i[\![s]\!])$

where α_i, i=1,2, is an *abstraction* operator which forgets some information present in $\mathcal{D}_i[\![s]\!]$. The proof of (1.2) requires an interesting technique of introducing a transition based *intermediate* semantics $\mathcal{I}_i[\![s]\!]$. For i=1 we shall show that $\mathcal{I}_i[\![s]\!] = \mathcal{D}_i[\![s]\!]$. Next, we introduce our first abstraction operator α_1 (turning each failing communication into an indication of failure and deleting all subsequent actions) and prove that $\mathcal{O}_1[\![s]\!] = \alpha_1(\mathcal{I}_1[\![s]\!])$.

The case i=2 is more involved, because $\mathcal{L}_1$ has *local*, and $\mathcal{L}_2$ *global* nondeterminacy. Consider a choice a *or* c, where a is some autonomous action and c needs a parallel $\bar{c}$ to communicate. In the case of local nondeterminacy (written as $a \cup c$) both actions may be chosen; in the global nondeterminacy case (written as $a+c$, + as in CCS [16]) c is chosen *only* when in some parallel compound $\bar{c}$ is ready to execute. Therefore, $\mathcal{L}_1$ and $\mathcal{L}_2$ exhibit different deadlock behaviours. $\mathcal{O}_2$ is based on the transition system T_2 which is a refinement of T_1, embodying a more subtle set of rules to deal with nondeterminacy. The denotational semantics $\mathcal{D}_2$ is as in [3,4,5]. In order to relate $\mathcal{D}_2$ and $\mathcal{O}_2$ we introduce the notion of *readies* and associated intermediate semantics $\mathcal{I}_2$, inspired by ideas as described in [8,18,19,21]. $\mathcal{I}_2$ involves an extension of the LT model with *some* branching information (though less than the full BT model) which is amenable to a treatment in terms of transitions. The proof of the desired result is then obtained by relating the semantics $\mathcal{O}_2$, $\mathcal{D}_2$ and $\mathcal{I}_2$ by a careful choice of suitable abstraction operators.

As main contributions of our paper we see

1. The three transition systems T_i, in particular the refinement of T_1 into T_2.
2. The systematic treatment of the denotational semantics definitions (for the guarded case) together with the settling of the relationship $\mathcal{O}_i = \alpha_i \circ \mathcal{D}_i$. ($\alpha_0$ identity).
3. Clarification of local versus global nondeterminacy and associated deadlock behaviour.
4. The intermediate semantics $\mathcal{I}_1$ and, in particular, $\mathcal{I}_2$.

2. THE LANGUAGE $\mathcal{L}_0$: SHUFFLE AND LOCAL NONDETERMINACY

Let A be a finite alphabet of elementary actions with $a \in A$. Let x,y be elements of the alphabet *Stmv* of statement variables (used in fixed point constructs for recursion). As syntax for $s \in \mathcal{L}_0$ we give

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

A term $\mu x[s]$ is a recursive statement. For example, according to the definitions to be proposed presently, the intended meaning of $\mu x[(a;x) \cup b]$ is the set $\{a^\omega\} \cup a^*.b$, with a the infinite sequence of a's.

2.1. *The transition system* T_0

Let $A^{tr} \overset{df.}{=} A^* \cup A^\omega \cup A^*.\{\bot\}$, with A^* the set of all finite words over A, $A^*.\{\bot\}$ the set of all (finite) unfinished words over A, and A^ω the set of all infinite words over A, and $\bot \notin A$. Let w,u,v denote elements of A^{tr}, and let λ be the empty word. We define $\bot.w = \bot$ for all w.

A *configuration* is a pair <s,w> or just a word w. A *transition relation* is a binary relation over configurations. A *transition* is a formula <s,w> $\rightarrow$ <s',w'> or <s,w> $\rightarrow$ w' denoting an element of a transition relation. A *transition system* is a formal deductive system for proving transitions based on *axioms* and *rules*. Using a self-explanatory notation, axioms have the format $1 \to 2$, rules have the format $\frac{1\to 2}{3\to 4}$. Also, $1 \to 2|3$ abbreviates $1 \to 2$ and $1 \to 3$, and $\frac{1\to 2|3}{4\to 5|6}$ abbreviates $\frac{1\to 2}{4\to 5}$ and $\frac{1\to 3}{4\to 6}$. For a transition system T, $T \vdash (1 \to 2)$ expresses that transition $1 \to 2$ is deducible from system T.

We now present the transition system T_0 for L_0:

$\langle s,w\rangle \to w$, $w \in A^{\omega} \cup A^*.\{\perp\}$. For $w \in A^*$ we put

(elementary action)

$\langle a,w\rangle \to w.a$

(local nondeterminacy)

$\langle s_1 \cup s_2,w\rangle \to \langle s_1,w\rangle \mid \langle s_2,w\rangle$

(recursion)

$\langle \mu x[s],w\rangle \to \langle s[\mu x[s]/x],w\rangle$

where, in general, $s[t/x]$ denotes substitution of t for x in s

(sequential composition)

$$\frac{\langle s_1,w_1\rangle \to w' \mid \langle s',w'\rangle}{\langle s_1;s_2,w_1\rangle \to \langle s_2,w'\rangle \mid \langle s';s_2,w'\rangle}$$

(shuffle)

$$\frac{\langle s_1,w_1\rangle \to w' \mid \langle s',w'\rangle}{\langle s_1 \| s_2,w_1\rangle \to \langle s_2,w'\rangle \mid \langle s' \| s_2,w'\rangle}$$

$$\frac{\langle s_1,w_1\rangle \to w' \mid \langle s',w'\rangle}{\langle s_2 \| s_1,w_1\rangle \to \langle s_2,w'\rangle \mid \langle s_2 \| s',w'\rangle}$$

2.2. *The operational semantics* $\mathcal{O}_0$

We show how to obtain $\mathcal{O}_0$ from T_0. We define the set $\mathcal{O}_0[\![s]\!]$ by putting $w \in \mathcal{O}_0[\![s]\!]$ iff one of the following three conditions is satisfied (always taking $\langle s_0,w_0\rangle =^{df.} \langle s,\lambda\rangle$):

1. There is a finite sequence of T_0-transitions
 $\langle s_0,w_0\rangle \to \dots \to \langle s_n,w_n\rangle \to w$
2. There is an infinite sequence of T_0-transitions
 $\langle s_0,w_0\rangle \to \dots \to \langle s_n,w_n\rangle \to \langle s_{n+1},w_{n+1}\rangle \to \dots$
 where the sequence $\langle w_n\rangle_{n=0}^{\infty}$ is infinitely often increasing, and $w = \sup_n w_n$ (sup with respect to prefix ordering).
3. There is an infinite sequence as in 2, but now
 $w_{n+k} = w_n$ for some n and all $k \geq 0$ and $w = w_n.\perp$

Examples. $\mathcal{O}_0[\![(a_1;a_2) \| a_3]\!] = \{a_1a_2a_3, a_1a_3a_2, a_3a_1a_2\}$, $\mathcal{O}_0[\![\mu x[(a;x) \cup b]]\!] = a^*.b \cup \{a^{\omega}\}$, $\mathcal{O}_0[\![\mu x[(x;a) \cup b]]\!] = b.a^* \cup \{\perp\}$.

Remark: Observe that systems such as T_0 are used to deduce (one step) transitions $1 \to 2$. *Sequences* of such transitions are used only to define $\mathcal{O}_0[\![.]\!]$.

2.3. *The denotational semantics* $\mathcal{D}_0$

We introduce a denotational semantics $\mathcal{D}_0$ for the language L_0 based on an approach using metric spaces (rather than the more customary cpo's) as underlying structure. This section is based on [3]; for the topology see [10]. We recall that $\mathcal{D}_i$ is defined only for the guarded case: Each $\mu x[s]$ is such that all free occurrences of x in s are sequentially preceded by some statement.

For $u \in A^{tr}$ let $u[n]$, $n \geq 0$, be the prefix of u of length n if this exists, otherwise $u[n] = u$. E.g., $abc[2] = ab$, $abc[5] = abc$. We define a natural metric d on A^{tr} by putting

$$d(u,v) = 2^{-\max\{n \mid u[n] = v[n]\}}$$

with the understanding that $2^{-\infty} = 0$. For example, $d(abc,abd) = 2^{-2}$, $d(a^n,a^{\omega}) = 2^{-n}$. We have that (A^{tr},d) is a complete metric space. For $X \subseteq A^{tr}$ we put $X[n] = \{u[n] \mid u \in X\}$. A distance $\hat{d}$ on subsets X,Y of A^{tr} is defined by

$$\hat{d}(X,Y) = 2^{-\max\{n \mid X[n] = Y[n]\}}$$

Let $\mathcal{C}$ denote the collection of all *closed* subsets of A^{tr}. It can be shown that $(\mathcal{C},\hat{d})$ is a complete metric space. A sequence $\langle X_i\rangle_{i=0}^{\infty}$ of elements of $\mathcal{C}$ is a *Cauchy sequence* whenever $\forall \varepsilon > 0\ \exists N\ \forall n,m \geq N[\hat{d}(X_n,X_m) < \varepsilon]$. For $\langle X_i\rangle_i$ a Cauchy sequence, we write $\lim_i X_i$ for its limit (which belongs to $\mathcal{C}$ by the completeness property).

A function ϕ: $(\mathcal{C},\hat{d}) \to (\mathcal{C},\hat{d})$ is called *contracting* whenever, for all X,Y, $\hat{d}(\phi(X),\phi(Y)) \le \alpha \cdot \hat{d}(X,Y)$, for some real number α with $0 \le \alpha < 1$. A classical theorem due to Banach states that in any complete metric space, a contracting function has a unique fixed point obtained as $\lim_i \phi^i(X_0)$ for arbitrary starting point X_0.

We now define the operations $.,\cup,\|$ on $\mathcal{C}$ in the following way:

a. $X,Y \subseteq A^* \cup A^*.\{\perp\}$. For X.Y and $X \cup Y$ we adopt the usual definitions (including the clause $\perp.u = \perp$ for all u). For $X \| Y$ we introduce as auxiliary operator the so-called left-merge $⌊\!⌊$ (from [7]). We put $X \| Y = (X ⌊\!⌊ Y) \cup (Y ⌊\!⌊ X)$, where $⌊\!⌊$ is given by $X ⌊\!⌊ Y = \bigcup\{u ⌊\!⌊ Y \mid u \in X\}$, $\varepsilon ⌊\!⌊ Y = Y$, $a ⌊\!⌊ Y = a.Y$, $\perp ⌊\!⌊ Y = \{\perp\}$, and $(a.u) ⌊\!⌊ Y = a.(\{u\} \| Y)$.

b. $X,Y \in \mathcal{C}$, X.Y do not consist of finite words only. Then $X \ op\ Y = \lim_i (X[i] \ op\ Y[i])$, for $op \in \{.,\cup,\|\}$. In [3] we have shown that this definition is well-formed and preserves closed sets, and the operations are continuous (for this finiteness of A is necessary).

We proceed with the definition of $\mathcal{D}_0[\![s]\!]$ for $s \in \mathcal{L}_0$. We introduce the usual notion of *environment* which is used to store and retrieve meanings of statement variables. Let $\Gamma = Stmv \to \mathcal{C}$ be the set of environments, and let $\gamma \in \Gamma$. We write $\gamma' =^{df.} \gamma\langle X/x\rangle$ for a *variant* of γ which is like γ but such that $\gamma'(x) = X$. We define $\mathcal{D}_0$: $\mathcal{L}_0 \to (\Gamma \to \mathcal{C})$ as follows:

DEFINITION.

$\mathcal{D}_0[\![a]\!](\gamma) = \{a\}$, $\mathcal{D}_0[\![s_1 \ op\ s_2]\!](\gamma) = \mathcal{D}_0[\![s_1]\!](\gamma) \ op\ \mathcal{D}_0[\![s_2]\!](\gamma)$, for $op \in \{.,\cup,\|\}$, $\mathcal{D}_0[\![x]\!](\gamma) = \gamma(x)$, and

$\mathcal{D}_0[\![\mu x[s]]\!](\gamma) = \lim_i X_i$, where $X_0 = \{\perp\}$ and $X_{i+1} = \mathcal{D}_0[\![s]\!](\gamma\langle X_i/x\rangle)$

By the guardedness requirement, each function $\phi = \lambda X.\ \mathcal{D}_0[\![s]\!](\gamma\langle X/x\rangle)$ is contracting, $\langle X_i\rangle_i$ is a Cauchy sequence, and $\lim_i X_i$ equals the unique fixed point of ϕ.

Remark. An order-theoretic approach to the denotational model is also possible (cf. [9,15]). However, for our present purposes this has no special advantages. In fact, the order-theoretic approach does not provide a *direct* treatment for the unguarded case either, it seems to require a contractivity argument for uniqueness of fixed points just as well, and, last but not least, as far as we know, it cannot be used as a basis for the BT model.

2.4. *Relationship between $\mathcal{O}_0$ and $\mathcal{D}_0$.*

We shall prove (for statements s without free statement variables, and omitting γ).

THEOREM 2.1. $\mathcal{O}_0 = \mathcal{D}_0$.

The proof relies on four lemmas.

LEMMA 2.2. $\mathcal{O}_0$ is homomorphic over $.,\cup,\|$.

LEMMA 2.3. (guarded case only). Consider a μ-term $\mu x[s]$. Let Ω be the (auxiliary) statement such that $\langle\Omega,w\rangle \to w.\perp$. Let $s^{(0)} = \Omega$, $s^{(n+1)} = s[s^{(n)}/x]$. Then $\mathcal{O}_0[\![\mu x[s]]\!] = \lim_n \mathcal{O}_0[\![s^{(n)}]\!]$.

PROOF. This involves a detailed analysis of transition sequences; it introduces in particular the notion of *truncating* a sequence after n applications of the recursion axiom involving the considered μ-term.

LEMMA 2.4. (guarded case only). For each s, $\mathcal{O}_0[\![s]\!]$ is a closed set.

Caution. This is not true for the unguarded case. For example, $\mathcal{O}_0[\![\mu x[(x;a) \cup b]]\!] = \{\perp\} \cup b.a^*$. This set is not closed since its limit point ba^ω is not in it.

LEMMA 2.5. (this is the crucial lemma relating $\mathcal{O}_0$

and $\mathcal{D}_0$). Let $var(s) \subseteq \{x_1,\ldots,x_n\}$. Let t_i be without free statement variables, and let $X_i = \mathcal{O}_0[\![t_i]\!]$, $i=1,\ldots,n$. Then

$$\mathcal{D}_0[\![s]\!](\gamma\langle X_i/x_i\rangle_{i=1}^n) = \mathcal{O}_0[\![s\langle t_i/x_i\rangle_{i=1}^n]\!].$$

PROOF. Structural induction on s.

3. THE LANGUAGE L_1: SYNCHRONIZATION MERGE AND LOCAL NONDETERMINACY

Let A be a finite alphabet, let $C \subseteq A$ with $c \in C$ (the *communications*) and let $a \in A\backslash C$. Let there be given a bijection $\bar{\ }: C \to C$ (*matching* communications à la CCS/CSP) with $\bar{\bar{c}} = c$. Let $\tau \in A$ be a special symbol serving as a meaning for the skip statement, and let δ be an element not in A indicating failure. We always have $\delta.w = \delta$. Let

$$A_\delta^{tr} = A^* \cup A^\omega \cup A^*.\{\delta,\perp\}$$

u,v,w now range over A_δ^{tr}. As syntax for $s \in L_1$ we give

$$s ::= a \mid c \mid \underline{skip} \mid \underline{fail} \mid s_1;s_2 \mid s_1 \cup s_2 \mid s_1 \| s_2 \mid x \mid \mu x[s].$$

3.1. *The transition system* T_1.

The system T_1 consists of T_0 extended with: $\langle s,w\rangle \to w$ for $w \in A^\omega \cup A^*.\{\delta,\perp\}$. For $w \in A^*$ we have

(communication)

$\langle c,w\rangle \to \langle\underline{fail},w\rangle$ an individual communication fails

(skip)

$\langle\underline{skip},w\rangle \to w.\tau$

(failure)

$\langle\underline{fail},w\rangle \to w.\delta$

(synchronization)

$\langle c \| \bar{c},w\rangle \to \langle\underline{skip},w\rangle$

$\langle c;s_1 \| \bar{c},w\rangle \to \langle\underline{skip};s_1,w\rangle$

$\langle c \| \bar{c};s_2,w\rangle \to \langle\underline{skip};s_2,w\rangle$

$\langle c;s_1 \| \bar{c};s_2,w\rangle \to \langle\underline{skip};(s_1 \| s_2),w\rangle$

(commutativity and associativity of merge)

$$\frac{\langle s_1 \| s_2,w\rangle \to \langle s',w'\rangle}{\langle s_2 \| s_1,w\rangle \to \langle s',w'\rangle}$$

$$\frac{\langle s_1 \| (s_2 \| s_3),w\rangle \to \langle s',w'\rangle}{\langle (s_1 \| s_2) \| s_3,w\rangle \to \langle s',w'\rangle}\ \text{, and symmetric.}$$

Remark. Note that associativity/commutativity of merge are *provable* in T_0.

3.2. *The operational semantics* $\mathcal{O}_1$

$\mathcal{O}_1[\![s]\!]$ is defined similarly to $\mathcal{O}_0[\![s]\!]$. Now failing communications result in δ, successful communications (through the synchronization rule) in addition in τ.

Examples. $\mathcal{O}_1[\![c]\!] = \{\delta\}$, $\mathcal{O}_1[\![(a;b) \cup (a;c)]\!] = \{ab,a\delta\}$, $\mathcal{O}_1[\![c \| \bar{c}]\!] = \{\delta,\tau\}$. We observe too many δ's here: to do away with such appearances of deadlocks in case an alternative is present, we postulate - *for the remainder of section 3 only* - the axiom

(3.1) $\{\delta\} \cup X = X$, for $X \neq \emptyset$

(Formally, we should now take congruence classes in A^{tr} with respect to (3.1); we do not bother to be that precise.) Taking (3.1) into account, the above examples now become $\mathcal{O}_1[\![c]\!] = \{\delta\}$, $\mathcal{O}_1[\![(a;b) \cup (a;c)]\!] = \{ab\}$, $\mathcal{O}_1[\![c \| \bar{c}]\!] = \{\tau\}$.

It is important to observe that the two statements $(a;b) \cup (a;c)$ and $a;(b \cup c)$ obtain the same meaning by $\mathcal{O}_1$. Section 4 will provide a more refined treatment.

3.3. *The denotational semantics* $\mathcal{D}_1$.

This is as in section 2,3. but extended/modified in the following way (omitting γ-arguments for simplicity):

$\mathcal{D}_1[\![c]\!] = \{c\}$, $\mathcal{D}_1[\![\underline{skip}]\!] = \{\tau\}$, $\mathcal{D}_1[\![\underline{fail}]\!] = \{\delta\}$, $\mathcal{D}_1[\![s_1 \| s_2]\!] = \mathcal{D}_1[\![s_1]\!] \| \mathcal{D}_1[\![s_2]\!]$, where, for $X,Y \subseteq A^{tr}$, we define $X \| Y = (X \mathbin{\|\!\_} Y) \cup (Y \mathbin{\|\!\_} X) \cup (X|Y)$. Here the operations $\mathbin{\|\!\_}$ (left-merge) and $|$ (communication) are defined as follows: First we take the case that X,Y consist of finite words only.

$X \mathbin{⫫} Y = \bigcup\{w \mathbin{⫫} Y \mid w \in X\}$, $\bot \mathbin{⫫} Y = \{\bot\}$, $\delta \mathbin{⫫} Y = \{\delta\}$, $\varepsilon \mathbin{⫫} Y = Y$, $a \mathbin{⫫} Y = a.Y$, $(a.w) \mathbin{⫫} Y = a(\{w\} \| Y)$. Also, $X|Y = \{(w|u): w \in X, u \in Y\}$, where $(c.w_1)|(\bar{c}.u_1) = \tau.(w_1 \| u_1)$. Moreover, $w'|u' = \delta$ for w',u' not of such a form. If X or Y contains infinite words, the definition is completed by taking limits. (The definition of $X \| Y$ is from [7].)

3.4. *Relationship between $\mathcal{O}_1$ and $\mathcal{D}_1$.*

We do not simply have that

(*) $\mathcal{O}_1[\![s]\!] = \mathcal{D}_1[\![s]\!]$.

(Take $s=c$ for a counter example. Then $\mathcal{O}_1[\![c]\!] = \{\delta\}$, $\mathcal{D}_1[\![c]\!] = \{c\}$). We even have that:

THEOREM 3.1. There does not exist any denotational (implying *compositional*) semantics $\mathcal{D}$ satisfying (*).

The proof is based on

LEMMA 3.2. $\mathcal{O}_1$ does not behave compositionally over $\|$.

Proof. We show that there exists no "mathematical" operator $\|_{\mathcal{D}}$ such that $\mathcal{O}_1[\![s_1 \| s_2]\!] = \mathcal{O}_1[\![s_1]\!] \|_{\mathcal{D}} \mathcal{O}_1[\![s_2]\!]$. Consider the programs $s_1 = c$, $s_2 = \bar{c}$ in $\mathcal{L}_1$. Then $\mathcal{O}_1[\![s_1]\!] = \mathcal{O}_1[\![s_2]\!] = \delta$. Suppose now that $\|_{\mathcal{D}}$ exists. Then $\{\delta\} = \mathcal{O}[\![s_1 \| s_1]\!] = \mathcal{O}[\![s_1]\!] \|_{\mathcal{D}} \mathcal{O}[\![s_1]\!] = \mathcal{O}[\![s_1]\!] \|_{\mathcal{D}} \mathcal{O}[\![s_2]\!] = \mathcal{O}[\![s_1 \| s_2]\!] = \{\tau\}$. Contradiction. □

We remedy this not by redefining T_1 (which adequately captures the operational intuition for $\mathcal{L}_1$), but rather by introducing an *abstraction* mapping α_1 such that

(**) $\mathcal{O}_1 = \alpha_1 \circ \mathcal{D}_1$.

We take $\alpha_1 = syn_1$ defined by $(W \subseteq A_\delta^{tr})$

$$syn_1(W) = \{w \mid w \in W \text{ does not contain } c \in C\} \cup \{w.\delta \mid \exists w',c' \text{ such that } w.c'.w' \in W, w \text{ contains no } c\}$$

The right-hand side of this definition should be taken with respect to ($\delta.w = \delta$ and) $\{\delta\} \cup X = X$, $X \neq \emptyset$. Informally, syn_1 replaces unsuccessful synchronization by deadlock and keeps this in case there is no alternative.

We cannot prove (**)' by a direct structural induction on s (because syn_1 does not behave homomorphically). Rather, we introduce an intermediate semantics $\mathcal{I}_1$: we modify T_1 into T_1^* which is the same as T_1 but for the communication axiom which now has the form

(communication*)

$\langle c,w\rangle \to w.c$

We base $\mathcal{I}_1$ on T_1^* just as we based $\mathcal{O}_1$ on T_1. We can now prove

LEMMA 3.3. For all $s,s' \in \mathcal{L}_1$ and $w,w' \in (A\backslash C)^*$

$T_1 \vdash \langle s,w\rangle \to w' \mid \langle s',w'\rangle$

iff

$T_1^* \vdash \langle s,w\rangle \to w' \mid \langle s',w'\rangle$

Proof. Structural induction on the deductions in T_1 and T_1^*. □

This lemma immediately leads to

THEOREM 3.4. $\mathcal{O}_1[\![s]\!] = syn_1(\mathcal{I}_1[\![s]\!])$

Next we show

THEOREM 3.5. $\mathcal{I}_1[\![s]\!] = \mathcal{D}_1[\![s]\!]$

Proof. Combine ideas of section 2.4 with a proof that $\mathcal{I}_1$ behaves compositionally over $\|$ (as defined in section 3.3).

Remark. This proof recalls Apt's merging lemma [1,2].

By combining theorems 3.4, 3.5 we finally obtain our desired result

THEOREM 3.6. $\mathcal{O}_1[\![s]\!] = syn_1(\mathcal{D}_1[\![s]\!])$.

4. THE LANGUAGE $\mathcal{L}_2$: SYNCHRONIZATION MERGE AND GLOBAL NONDETERMINACY

The syntax for $s \in \mathcal{L}_2$ is given by

$s ::= a \mid c \mid \underline{skip} \mid \underline{fail} \mid s_1;s_2 \mid s_1+s_2 \mid s_1 \| s_2 \mid x \mid \mu x[s]$

Here "+" denotes global nondeterminacy; the notation is from CCS[16].

4.1. *The transition system* T_2.

T_2 is like T_1, but without the axiom for local nondeterminacy, and without the axiom for communication ($<c,w> \to <\underline{fail},w>$). Additionally, we have

(global nondeterminacy)

[μ-unfolding]

$$\frac{<s_1,w> \to <s',w>}{<s_1+s_2,w> \to <s'+s_2,w>}$$

[selection by elementary action]

$$\frac{<s_1,w> \to w' \mid <s',w'>}{<s_1+s_2,w> \to w' \mid <s',w'>}, \text{ where } w' \neq w$$

[selection by communication/synchronization]

$$\frac{<s_1 \| s_3,w> \to <s',w'>}{<(s_1+s_2) \| s_3,w> \to <s',w'>}, \text{ where the}$$

transition in the premise involves synchronization between actions from s_1 and s_3

[commutativity of +]

$$\frac{<s_1+s_2,w> \to w' \mid <s',w'>}{<s_2+s_1,w> \to w' \mid <s',w'>}$$

$$\frac{<(s_1+s_2) \| s_3,w> \to w' \mid <s',w'>}{<(s_2+s_1) \| s_3,w> \to w' \mid <s',w'>}$$

Remark. Associativity of + is derivable.

We see that global nondeterminacy is more restrictive than local nondeterminacy. In fact, $<s_1+s_2,w> \to w' \mid <s',w'>$ implies $<s_1 \cup s_2,w> \to w' \mid <s',w'>$ but not vice versa.

Example. $<a \cup c,w> \to^* w.\delta, <a \cup c,w> \to^* w.a$, but $<a+c,w> \to^* w.a$ only. In the case of global nondeterminacy, the communication transitions of s_1+s_2 depend on the communication transitions of s_1 and s_2 in some global context $s_1 \| s_3$ or $s_2 \| s_3$.

This formalizes the communication as present in languages like CSP, ADA or OCCAM.

4.2. *The operational semantics* $\mathcal{O}_2$

$\mathcal{O}_2$ is derived form T_2 in the usual way. In addition, however, we now have to consider the case that we have a finite sequence $<s,\lambda> = <s_0,w_0> \to \ldots \to <s_n,w_n>$, with no transition $<s_n,w_n> \to \ldots$ deducible. We then deliver $w_n.\delta$ as element of $\mathcal{O}_2[\![s]\!]$. The pair $<s_n,w_n>$ is then called a deadlocking configuration.

Example. $\mathcal{O}_2[\![(a;b)+(a;c)]\!] = \{ab,a\delta\}$, $\mathcal{O}_2[\![a;(b+c)]\!] = \{ab\}$.

4.3. *The denotational semantics* $\mathcal{D}_2$.

We follow [3,4,5] in introducing a branching time semantics for L_2. Let $A_\perp \overset{df.}{=} A \cup \{\perp\}$. Let P_n, $n \geq 0$, be defined by

$$P_0 = \mathcal{P}(A_\perp),\ P_{n+1} = \mathcal{P}(A_\perp \cup (A_\perp \times P_n))$$

where $\mathcal{P}(.)$ denotes all subsets of (.), and let $P_\omega = \bigcup_n P_n$. We define a metric d on P_ω (for its definition see [3,4,5]) and take P as the *completion* of P_ω with respect to d. It can be shown that P satisfies the domain equation

$$P = \mathcal{P}_{closed}(A_\perp \cup (A_\perp \times P))$$

Finite elements of P are, e.g., $\{[a,\{b_1\}],[a,\{b_2\}]\}$ or $\{[a,\{b_1,b_2\}]\}$. Thus, the branching structure is preserved. An infinite element is, e.g., the process p which satisfies the equation $p = \{[a,p],[b,p]\}$. The empty set *is* a process and takes the role of δ. Note that in the LT framework, ∅ cannot replace δ since by the definition of concatenation (for LT) we have $a.\emptyset = \emptyset$ which is undesirable for an element modelling failure. (An action which fails should not cancel all previous actions.) In the BT framework, $\{[a,\emptyset]\}$ is a process which is indeed different from ∅. Since, clearly, $\emptyset \cup p = p$ for all sets (processes) p, we can do without explicitly imposing a counterpart of rule (3.1) for δ.

Operations $.,\cup,\|$, limits and continuity, fixed points of contracting operations are as in [3,4,5]. For example, for $p,q \in P_\omega$, we put $p\| q = (p\|\!\_ q) \cup (q\|\!\_ p) \cup (p|q)$ where $p\|\!\_ q = \{x\|\!\_ q: x \in p\}$, $a\|\!\_ q = [a,q], \bot\|\!\_ q = \bot$, $[a,p']\|\!\_ q = [a,p'\| q]$, and $p|q = \bigcup\{(x|y): x \in p, y \in q\}$, where $[c,p']|[\bar{c},q'] = \{[\tau,p'\| q']\}$, $c|[\bar{c},q'] = \{[\tau,q']\}$, $[c,p']|\bar{c} = \{[\tau,p']\}$, $c|\bar{c} = \{\tau\}$, and $(x|y) = \emptyset$ when x,y are not of one of these four forms.

It is now straightforward to define $\mathcal{D}_2: L_2 \to (\Gamma_2 \to P)$, where $\Gamma_2 = Stmv \to P$, by following the clauses in the definition of $\mathcal{D}_0$, $\mathcal{D}_1$. Thus we put $\mathcal{D}_2[\![a]\!](\gamma) = \{a\}$, $\mathcal{D}_2[\![s_1 \; op \; s_2]\!](\gamma) = \mathcal{D}_2[\![s_1]\!](\gamma) \; op \; \mathcal{D}_2[\![s_2]\!](\gamma)$, $\mathcal{D}_2[\![x]\!](\gamma) = \gamma(x)$, and

$\mathcal{D}_2[\![\mu x[s]]\!](\gamma) = \lim_i p_i$, where $p_0 = \{\bot\}$ and $p_{i+1} = \mathcal{D}_2[\![s]\!](\gamma\langle p_i/x\rangle)$

4.4. *Relationship between* $\mathcal{O}_2$ *and* $\mathcal{D}_2$.

We shall show that

(*) $\mathcal{O}_2 = \alpha_2 \circ \mathcal{D}_2$,

for suitable α_2. In fact, α_2 is defined in two steps:

1. First we define $syn_2: P \to P$ for $p \in P_\omega$

   $syn_2(p) = \{a|\; a \in p \text{ and } a \notin C\} \cup \{[a,syn_2(q)]|\; [a,q] \in p \text{ and } a \notin C\}$

   For $p \in P \backslash P_\omega$, we have $p = \lim_n p_n$, with $p_n \in P_n$, and we put $syn_2(p) = \lim_n(syn_2(p_n))$.

   Example. Let $p = \mathcal{D}_2[\![(a+c) \| (b+\bar{c})]\!]$. Then $syn_2(p) = \{[a,\{b\}],[b,\{a\}],\tau\}$.

2. Next, we define $traces: P \to P(A_\delta^{tr})$ by (finite case only displayed):

   $traces(p) = \bigcup\{traces(x): x \in p\}$ if $p \neq \emptyset$

   $= \{\delta\}$ if $p = \emptyset$

where $traces(a) = \{a\}$, $traces([a,q]) = a.traces(q)$.

We now put

$$\alpha_2 \stackrel{df.}{=} traces \circ syn_2,$$

but we cannot (yet) prove (*), because, similarly to α_1, α_2 does not behave homomorphically. Therefore, we try an intermediate semantics $\mathcal{I}_2$. This cannot be based on a simple LT model as the following argument shows:

Let us try for $\mathcal{I}_2$, similarly to $\mathcal{I}_1$, the addition of the axiom $\langle c,w\rangle \to w.c$ to T_2. Now consider the programs $s_1 \equiv a;(c_1+c_2)$, $s_2 \equiv (a;c_1) + (a;c_2)$, $s \equiv \bar{c}_1$. Then $\mathcal{O}_2[\![s_1 \| s]\!] = \{a\tau\} \neq \{a\tau,a\delta\} = \mathcal{O}_2[\![s_2 \| s]\!]$. However, $\mathcal{I}_2[\![s_1 \| s]\!] = \mathcal{I}_2[\![s_2 \| s]\!]$. Thus whatever α we apply to $\mathcal{I}_2[\![.]\!]$, the results for $s_1 \| s$, $s_2 \| s$ will turn out the same.

Our solution to this problem is to introduce an intermediate semantics $\mathcal{I}_2$ which, besides recording all traces in A_δ^{tr}, also records a very weak information about the *local branching structure* of the process. This information is called a *ready set* or *deadlock possibility*: it is a subset X of C. Informally, X indicates the set of communications c which are ready to synchronize with any other matching communication $\bar{c}$ from another parallel compound (for the notion of *ready set* cf. [8,11,18,19,21]). Formally, take $\Delta = P(C)$. For $X \in \Delta$, let $\bar{X} = \{\bar{c}|\; c \in X\}$. The *ready* domain $\mathcal{R}$ is now $\mathcal{R} = P(A^{tr} \cup A^{tr}.\Delta)$. The transition system T_2^* consists of all axioms and rules of T_2 together with (for $w \in A^*$).

(i) $\langle c,w\rangle \to w.c$

(ii) $\langle c,w\rangle \to w.\{c\}$

(iii) $\langle \underline{fail},w\rangle \to w.\emptyset$

(iv) $$\frac{\langle s_1,w\rangle \to w.X \quad , \quad \langle s_2,w\rangle \to w.Y}{\langle s_1+s_2,w\rangle \to w.X\cup Y}$$

(v) $$\frac{\langle s_1,w\rangle \to w.X \quad , \quad \langle s_2,w\rangle \to w.Y}{\langle s_1 \| s_2,w\rangle \to w.X\cup Y}$$, where $X\cap\bar{Y} = \emptyset$.

Axioms (ii),(iii) introduce deadlock possibilities/ready sets. Rule (iv) says that s_1+s_2 has a (one-step) deadlock possibility only if s_1 and s_2 have, and rule (v) says that $s_1 \| s_2$ has a (one-step) deadlock possibility if both s_1 and s_2 have, and no synchronization is possible. We omit the natural definition of $\mathcal{I}_2$ from T_2^*.

Examples ($\mathcal{I}_2$ semantics)

(i) $\mathcal{I}_2[\![a;(b+c)]\!] = \{ab,ac\}$.

Proof. We explore all transition sequences in T_2^* starting in $\langle a;(b+c),\lambda\rangle$:

(1)	$\langle a,\lambda\rangle \to a$	(elem.action)
(2)	$\langle a;(b+c),\lambda\rangle \to \langle b+c,a\rangle$	(seq.comp.:(1))
(3)	$\langle b.a\rangle \to ab$	(elem.action)
(4)	$\langle c,a\rangle \to ac$, $\searrow a.\{c\}$	(comm.)
(5)	$\langle b+c.a\rangle \to ab$, $\searrow ac$	(glob.nondet.: (3),(4))

No more transitions are deducible for $\langle b+c,a\rangle$.

(6) Thus

$\langle a;(b+c),\lambda\rangle \to \langle b+c,a\rangle \to ab$, $\searrow ac$

are all transition sequences starting in $\langle a;(b+c),\lambda\rangle$.

This proves the claim □

(ii) $\mathcal{I}_2[\![a;b + a;c]\!] = \{ab,ac,a.\{c\}\}$.

Proof. Here we only exhibit all possible transition sequences in T_2^* starting in $\langle a;(b+c),\lambda\rangle$:

$\langle a;b+a;c,\lambda\rangle \to \langle b,a\rangle \to ab$
$\searrow \langle c,a\rangle \to ac$
$\searrow a.\{c\}$ □

For the further results the following lemma is important:

LEMMA 4.1. For all $s,s' \in (A\backslash C)^*$ the following holds:

1. $T_2 \vdash \langle s,w\rangle \to w' | \langle s',w'\rangle$ iff $T_2^* \vdash \langle s,w\rangle \to w' | \langle s',w'\rangle$
2. $\langle s,w\rangle$ is a deadlocking configuration for T_2 iff there exists some $X \subseteq C$ with $T_2^* \vdash \langle s,w\rangle \to w.X$.

Let now w range over $A^{tr} = A^* \cup A^\omega \cup A^*.\{\perp\}$ and let W range over $\mathcal{R} = \mathcal{P}(A^{tr} \cup A^{tr}.\Delta)$. We define the abstraction operator syn_2^*: $\mathcal{R} \to \mathcal{P}(A_\delta^{tr})$ by

$syn_2^*(W) = \{w |\ w \in W$ does not contain any $c \in C\} \cup \{w\delta |\ \exists X \in \Delta: w.X \in W\}$

We have

THEOREM 4.2. $\mathcal{O}_2 = syn_2^* \circ \mathcal{I}_2$.

Next, we wish to relate $\mathcal{I}_2$ with the full BT semantics $\mathcal{D}_2$. To this end, we introduce the abstraction operator *readies*: $P \to \mathcal{R}$ by defining *readies*(p) as follows (finite case only). Let $p = \{a_1,\ldots,a_m,[b_1,q_1],\ldots,[b_n,q_n]\}$, with $a_i,b_j \in A$. We put

readies(p) $= \bigcup\{$*readies*$(x): x \in p\} \cup \{\lambda.X | X = \{a_1,\ldots,a_m,b_1,\ldots,b_n\} \subseteq C\}$

where *readies* $(a_i) = \{a_i\}$, *readies* $([b_j,q_j]) = b_j.$*readies* (q_j).

THEOREM 4.3. $\mathcal{I}_2 =$ *readies* $\circ\ \mathcal{D}_2$.

Proof. (i) *readies* behaves homomorphically on $.,+,\|$. (ii) $\mathcal{I}_2(\mu x[s])$ can be obtained by applying *readies* to the fixed point definition of $\mu x[s]$ under $\mathcal{D}_2$.

LEMMA 4.4. *traces* $\circ$ syn_2 = syn_2^* $\circ$ *readies*

Summarizing, we have our final

THEOREM 4.5. $\mathcal{O}_2$ = *traces* $\circ$ $syn_2 \circ \mathcal{D}_2$.

REFERENCES

[1] K.R. Apt (1981), Recursive Assertions and Parallel Programs, Acta Inf. 15, pp.219-232.

[2] K.R. Apt (1983), Formal Justification of a Proof System for Communicating Sequential Processes, J. Assoc. Comput. Mach.,30(1), pp.197-216.

[3] J.W. de Bakker, J.A. Bergstra, J.W. Klop & J.-J.Ch. Meyer (1983), Linear Time and Branching Time Semantics for Recursion with Merge, Theoretical Computer Science 34 (1984), pp.135-156.

[4] J.W. de Bakker & J.I. Zucker (1982), Denotational Semantics of Concurrency, in Proceedings 14th Assoc.Comput.Mach. Symp. on Computing, pp.153-158.

[5] J.W. de Bakker & J.I. Zucker (1982), Processes and the Denotational Semantics of Concurrency, Inform. and Control 54 (1/2), pp.70-120.

[6] J.W. de Bakker & J.I. Zucker (1983), Compactness in Semantics for Merge and Fair Merge, in: E. Clarke & D. Kozen (eds.), Proceedings Workshop Logics of Programs, Pittsburgh, Springer LNCS, 164, pp.18-33.

[7] J.A. Bergstra & J.W. Klop (1984), Process Algebra for Synchronous Communication, Information and Control 60 (1984),pp.109-137.

[8] S.D. Brookes, C.A.R. Hoare & A.W. Roscoe (1984), A Theory of Communicating Sequential Processes, J.Assoc.Comput. Mach.31(3), pp.560-599.

[9] M. Broy (1983), Fixed Point Theory for Communication and Concurrency, in: D. Bjørner (ed.), Proceedings IFIP Working Conference on Formal Description of Programming Concepts II, North-Holland Amsterdam, pp.125-148.

[10] J. Dugundji (1966), Topology, Allen and Bacon, Rockleigh, N.J.

[11] N. Francez, D.J. Lehmann & A. Pnueli (1984), A Linear-History Semantics for Languages for Distributed Programming, Theor.Comp. Sc. 32(1/2), pp.25-46.

[12] W.G. Golson & W.C. Rounds (1983), Connections between Two Theories of Concurrency: Metric Spaces and Synchronization Trees, Inform. and Control 57(2/3), pp.102-124.

[13] M. Hennessy & G.D. Plotkin (1979), Full Abstraction for a Simple Parallel Programming Language, in: J. Bečvař (ed.), Proceedings 8th MFCS, LNCS 74, Springer, Berlin/New York, pp.108-120.

[14] R. Keller (1976), Formal Verification of Parallel Programs, Comm. Assoc. Comput. Mach. 19, pp.371-384.

[15] J.-J.Ch. Meyer (1984), Fixed Points and the Arbitrary and Fair Merge of a Fairly Simple Class of Processes, Techn. Reports IR-89/IR-92. Free University, Amsterdam.

[16] R. Milner (1980), A Calculus for Communicating Systems, LNCS 92, Springer, Berlin/New York.

[17] M. Nivat (1979), Infinite Words, Infinite Trees, Infinite Computations, in Proceedings Found. of Comp. Sc. III.2, Mathematical Centre Tracts 109, Amsterdam, pp.3-52.

[18] E.-R. Olderog & C.A.R. Hoare (1983), Specification-oriented Semantics for Communicating Processes in: J. Diaz (ed.), Proceedings 10th Int. Coll. on Autom., Langu. and Programming, pp.561-572.

[19] E.-R. Olderog & C.A.R. Hoare (1984), Specification-oriented Semantics for Communicating Processes, Techn. Monograph PRG-37, Oxford Univ. Progr. Research Group.

[20] G.D. Plotkin (1983), An Operational Semantics for CSP in: D. Bjørner (ed.), Formal Description of Programming Concepts II, North-Holland, Amsterdam, pp.199-223.

[21] W.C. Rounds & S.D. Brookes (1981), Possible Futures, Acceptances, Refusals, and Communicating Processes, in Proceedings 22nd Symp. Found. of Comp. Sc., IEEE, pp.140-149.