
Transition systems, infinitary languages
and the semantics of uniform concurrency
Transition systems as proposed by Hennessy & Plotkin are defined for a series of three languages featuring concurrency. The first has shuffle and local nondeterminacy, 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 branching 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.
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:

$l_0$: shuffle (arbitrary interleaving) + local nondeterminacy (section 2)

$l_1$: synchronization merge + local nondeterminacy (section 3)

$l_2$: synchronization merge + global nondeterminacy (section 4)

For $l_i$ with typical elements $s$, we shall present transition system $T_i$ and define an induced operational semantics $O_i[s], i=0,1,2$. We shall also define three denotational semantics $D_i[s]$ 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 $D_i$ only for $l_i$ 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 $D_i$ do assign meaning to the unguarded case as well.)

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

$$O_i[s] = D_i[s]$$

We shall show that (1.1) only holds for $i=0$. For the more sophisticated languages $l_i, i=1,2$, we cannot prove (1.1). In fact, we can even show that there exists no $D_i$ satisfying (1.1), $i=1,2$. Rather than trying to modify $O_i$ (thus spoiling its intuitive operational character) we propose to replace (1.1) by
(1.2) \( \varphi_1[\epsilon] = \varphi_1(\varphi_1[\epsilon]) \)

where \( \varphi_1, i=1,2 \), is an abstraction operator which
forgets some information present in \( \varphi_1[\epsilon] \). The
proof of (1.2) requires an interesting technique
of introducing a transition based intermediate
semantics \( I_i[\epsilon] \). For \( i=1 \) we shall show that
\( I_1[\epsilon] = \varphi_1[\epsilon] \). Next, we introduce our first
abstraction operator \( \varphi_1 \) (turning each failing
communication into an indication of failure and
deleting all subsequent actions) and prove that
\( \varphi_1[\epsilon] = \varphi_1(\varphi_1[\epsilon]) \).

The case \( i=2 \) is more involved because \( \varphi_2 \)
has local and \( \varphi_2 \) global nondeterminacy. Consider a
choice \( a \) or \( c \), where \( a \) is some autonomous action
and \( c \) needs a parallel \( c \) to communicate. In the
case of local nondeterminacy (written as \( a + c \))
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 \( c \) is ready to execute. Therefore, \( \varphi_1 \)
and \( \varphi_2 \) exhibit different deadlock behaviours. \( \varphi_2 \)

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 \( \varphi_2 \) is as in [3,4,5]. In
order to relate \( \varphi_2 \) and \( \varphi_2 \), we introduce the notion
of readings and associated intermediate semantics
\( \varphi_2 \), inspired by ideas as described in [8,18,19,21].
\( \varphi_2 \) involves an extension of the LT model with
some branching information (though less than the
full NT model) which is amenable to a treatment
in terms of transitions. The proof of the desired
result is then obtained by relating the semantics
\( \varphi_2, \varphi_2 \) and \( \varphi_2 \) by a careful choice of suitable
abstraction operators.

As main contributions of our paper we see

1. The three transition systems \( t_1 \), 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
\( \varphi_1 = \varphi_1[\varphi_1[\epsilon]) \).
3. Clarification of local versus global
nondeterminacy and associated deadlock
behaviour.
4. The intermediate semantics \( t_i \) and, in
particular, \( t_2 \).

2. THE LANGUAGE \( L_0 \): SHUFFLE AND LOCAL NONDETERMINACY

Let \( \alpha \) be a finite alphabet of elementary
actions with \( a \in A \). Let \( a, b \) be elements of the
alphabet \( STM \) of statement variables (used in
fixed point constructs for recursion). As syntax
for \( s \in L_0 \) we give
\[ s ::= s_1 \mid s_2 \mid s_1 \cdot s_2 \mid s_1 \uparrow \uparrow s_1 \| s_2 \| c \]...
based on axioms and rules. Using a self-explanatory notation, axioms have the format \( 1 + 2 \), rules have the format \( 1 + 2 \) \( \frac{3}{4} \). Also, \( 1 + 2 \) \( \frac{3}{4} \) abbreviates \( 1 + 3 \) and \( 1 + 3 \) \( \frac{4}{5} \) abbreviates \( 2 + 3 \) \( \frac{4}{5} \) \( \frac{6}{7} \). For a transition system \( T \), \( T \vdash (1 + 2) \) expresses that transition \( 1 + 2 \) is deducible from system \( T \).

We now present the transition system \( T_0 \) for \( L_0 \):

\[
<s, w> + w, \ w \in A^* \{, .}.
\]

For \( w \in A^* \) we put

(1) Elementary action

\[
<s, w> + w.a
\]

(2) Local nondeterminacy

\[
<s_1 \cup s_2, w> + <s_1, w>, <s_2, w>, <s_1; s_2, w>
\]

(3) Recursion

\[
<\mu x[s], w> + <s[\mu x[s]/x], w>
\]

where, in general, \( s[t/x] \) denotes substitution of \( t \) for \( x \) in \( s \).

(4) Sequential composition

\[
<s_1; s_2, w> + <s_1, w'>, <s_2, w'>, <s_1; s_2, w'>
\]

(5) Shuffle

\[
<s_1; s_2, w> + <s_1, w'>, <s_2, w'>, <s_1; s_2, w'>
\]

2.2. The operational semantics \( \Omega_0 \)

We show how to obtain \( \Omega_0 \) from \( T_0 \). We define the set \( \Omega_0[s] \) by putting \( w \in \Omega_0[s] \) iff one of the following three conditions is satisfied (always taking \( \Omega_0[\cdot] = df.<s, w> \)):

1. There is a finite sequence of \( T_0 \)-transitions

\[
<s_0, w_0> \rightarrow \cdots \rightarrow <s_n, w_n> \rightarrow w
\]

2. There is an infinite sequence of \( T_0 \)-transitions

\[
<s_0, w_0> \rightarrow \cdots \rightarrow <s_n, w_n> \rightarrow <s_{n+1}, w_{n+1}> \rightarrow \cdots
\]

where the sequence \( <w_n\>_{n=0}^{\infty} \) 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 \( w = w_n \).

\[
\Omega_0[[a_1; a_2]| a_3] = \{a_1, a_2, a_3, a_2 a_3 a_1, a_3 a_1 a_2, a_3 a_1 a_2 a_3\},
\]

\[
\Omega_0[[\mu x[a(x)], b]] = a^* b u (a^*), \Omega_0[[\mu x[x]; a b]] = b a^* u \{1\}.
\]

Remark: Observe that systems such as \( T_0 \) are used to deduce (one step) transitions \( 1 + 2 \). Sequences of such transitions are used only to define \( \Omega_0[\cdot] \).

2.3. The denotational semantics \( \Omega_0 \)

We introduce a denotational semantics \( \Omega_0 \) 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 \( \Omega_0[\cdot] \) 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 \( w \in A^* \), let \( w[n], n \geq 0 \), be the prefix of \( w \) of length \( n \) if this exists, otherwise \( w[n] = u \).


We define a natural metric \( d \) on \( A^* \) by putting

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

with the understanding that \( 2^{-\infty} = 0 \). For example,

\[
d(abc, ab) = 2^{-2}, \quad d(a^n, a^\infty) = 2^{-n}.
\]

We have that \( (A^*, d) \) is a complete metric space. For \( X \subseteq A^* \), we put

\[
X[n] = \{u[n] \mid u \in X\}.
\]

A distance \( d \) on subsets \( X, Y \) of \( A^* \) is defined by

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

Let \( C \) denote the collection of all closed subsets of \( A^* \).

It can be shown that \( (C, d) \) is a complete metric space. A sequence \( \langle X^*_n \rangle_{n=0}^{\infty} \) of elements of \( C \) is a Cauchy sequence whenever

\[
\forall \varepsilon > 0 \ \exists n, m \geq 0 \ : d(X^*_n, X^*_m) < \varepsilon.
\]

For \( \langle X^*_n \rangle \), a Cauchy sequence, we write \( \lim_{n \to \infty} X^*_n \) for its limit (which belongs to \( C \) by the completeness property).
A function $\phi: (C, \hat{d}) \rightarrow (C, \hat{d})$ is called contracting whenever, for all $X, Y$, $\hat{d}(\phi(X), \phi(Y)) \leq \alpha \cdot \hat{d}(X, Y)$, for some real number $\alpha$ with $0 \leq \alpha < 1$. 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 $\cdot, \cup$ on $C$ in the following way:

a. $X, Y \in A^* \cup A^*$, $a$. For $X, Y$ and $X \cup Y$ we adopt the usual definitions (including the clause $\perp \mu = \perp$ for all $\mu$). For $X \parallel Y$ we introduce an auxiliary operator so-called left-merge $\perp$ (from [7]). We put $X \parallel Y = (X \cup Y) \cup (Y \cup X)$, where $\perp$ is given by $X \parallel Y = U \cup (U \cup Y) \cup (U \cup X)$, $U = Y = Y$, $U = X \parallel Y$. Let $U = (a, u) \parallel Y = a.(u) \parallel Y$.

b. $X, Y \in C, X \parallel Y$ do not consist of finite words only. Then $X \parallel Y = \lim_i (X \parallel Y) \parallel Y(i)$, for $\parallel \in \{ \cdot, \cup, \parallel \}$, 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 $\mathcal{P}_0[\Delta]$ for $s \in \mathcal{P}_0$. We introduce the usual notion of environment which is used to store and retrieve meanings of statement variables. Let $\Gamma = Stm \rightarrow C$ be the set of environments, and let $\gamma \in \Gamma$. We write $\gamma^t = \{t\}$ for a variant of $\gamma$ which is like $\gamma$ but such that $\gamma^t(x) = X$. We define $\mathcal{P}_0: \Gamma \rightarrow (\Gamma \times C)$ as follows:

**DEFINITION.**

- $\mathcal{P}_0[\Delta](\gamma) = \{s\}$, $\mathcal{P}_0[\Delta] \cup \gamma \mu(s) \mathcal{P}_0(\gamma)$ op $\mathcal{P}_0[\Delta](\gamma)$, for $\mu \in \{ \cdot, \cup, \parallel \}$, $\mathcal{P}_0[\Delta](\gamma) = \Delta(\gamma)$, and $\mathcal{P}_0[\Delta](\gamma) = \lim_i X_1$, where $X_0 = \{s\}$ and $X_{i+1} = \mathcal{P}_0[\Delta](\gamma^t_{i+1})$

By the guardedness requirement, each function $\phi = \Delta_\gamma. \mathcal{P}_0[\Delta](\gamma^t_{i+1})$ is contracting, $\lim_i X_1$ is a Cauchy sequence, and $\lim_i X_1$ equals the unique fixed point of $\phi$.

**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 $\mathcal{P}_0$ and $\mathcal{P}_0$

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

**THEOREM 2.1.** $\mathcal{P}_0 = \mathcal{P}_0$.

The proof relies on four lemmas.

**LEMMA 2.2.** $\mathcal{P}_0$ is homomorphic over $\cdot, \cup, \parallel$.

**LEMMA 2.3.** (guarded case only). Consider a $\mu$-term $\Delta[\mu]$. Let $\omega$ be the (auxiliary) statement such that $\langle \omega, \omega \rangle = \perp$, we let $s[0] = \omega, s[n+1] = s[n](\gamma^t)$.

Then $\mathcal{P}_0[\Delta[\mu](s)] = \lim_i \mathcal{P}_0[\Delta[\mu](n)]$.

**PROOF.** This involves a detailed analysis of transition sequences; it introduces in particular the notion of truncating a sequence after a number of the recursion axiom involving the considered $\mu$-term.

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

**LEMMA 2.5.** (this is the crucial lemma relating $\mathcal{P}_0$...
and $\mathcal{P}_0$. Let $\text{var}(s) \subseteq \{x_1, \ldots, x_n\}$. Let $t_1$ be without free statement variables, and let $x_1 = \emptyset_0[tx_{1\ldots n}]$, $i=1, \ldots ,n$. Then 
$$
\mathcal{P}[\mathcal{E}s] (\gamma x_1/x_1', 1\ldots n) = \emptyset_0[ts_{1/1\ldots n}],
$$

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 $a \in A\setminus C$. Let there be given a bijection $C \leftrightarrow C$ (matching communications à la CCS/CSP) with $c = c$. Let $T \in A$ be a special symbol serving as a meaning for the skip statement, and let $\delta$ be an element not in $A$ indicating failure. We always have $\delta, \omega = \delta$. Let 
$$
A_{\delta}^T = A^* \cup \omega \cup A^* \cdot \delta \cdot l
$$
and $\omega, w$ range over $A_{\delta}^T$. As syntax for $s \in L_1$ we give 
$$
s := s|\text{skip}|\text{fail}|s_1, s_2|s_1 \cup s_2 | s_1 \parallel s_2 | x | w|x[s].
$$

3.1. The transition system $T_1$.

The system $T_1$ consists of $T_0$ extended with: 
$$
<s, \omega> \rightarrow w \text{ for } w \in A^* \cup \omega \cdot (\delta, l).
$$

(Communication)

$$
<c, \omega> \rightarrow <\text{fail}, \omega>
$$

An individual communication fails.

(failure)

$$
<\text{skip}, \omega> \rightarrow \omega \cdot T
$$

 Kommunikationsstörungen

(synchronization)

$$
<s, \parallel c, \omega> \rightarrow <\text{skip}, \omega>
$$

$$
<s_1, \parallel c, \omega> \rightarrow <\text{skip}, s_1, \omega>
$$

$$
<s, \parallel c, \omega> \rightarrow <\text{skip}, s_2, \omega>
$$

$$
<s_1, \parallel c, \omega> \rightarrow <\text{skip}, s_1 \parallel s_2, \omega>
$$

(Commutativity and associativity of merge)

$$
<s_1 \parallel s_2 \parallel s_3 > \rightarrow <s_1 \parallel s_2 , \parallel s_3 > \rightarrow <s_1 \parallel s_2 \parallel s_3 > \rightarrow <s_1 \parallel s_2 \parallel s_3 >
$$

$$
\text{and symmetric.}
$$

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

3.2. The operational semantics $0_1$

$0_1[\mathcal{E}s]$ is defined similarly to $0_0[\mathcal{E}s]$. Now failing communications result in $\delta$, successful communications (through the synchronization rule) in addition to $T$.

Examples. $0_1[\mathcal{E}c] = \{\delta\}, 0_1[\mathcal{E}(a;b) \cup (a;c)] = \{ab, \delta\}, 0_1[\mathcal{E}c \parallel \mathcal{C}c] = \{\delta, T\}$. We observe too many $\delta$'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

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

(For example, we would now take congruence classes in $A^*\parallel$ with respect to (3.1); we do not bother to be that precise.) Taking (3.1) into account, the above examples now become $0_1[\mathcal{E}c] = \{\delta\}, 0_1[\mathcal{E}(a;b) \cup (a;c)] = \{ab\}, 0_1[\mathcal{E}c \parallel \mathcal{C}c] = \{T\}$.

It is important to observe that the two statements $(a;b) \cup (a;c)$ and $a\parallel (b;c)$ obtain the same meaning by $0_1$. Section 4 will provide a more refined treatment.

3.3. The denotational semantics $D_1$.

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

$$
D_1[\mathcal{E}c] = \{c\}, D_1[\text{skip}] = \{T\}, D_1[\text{fail}] = \{\delta\},
$$

$$
D_1[\mathcal{E}s_1 \parallel s_2] = D_1[\mathcal{E}s_1] \parallel D_1[\mathcal{E}s_2], \text{ where for } x,y \in A^*,
$$

we define $x || y = (xL \cup yL)U (yL \cup xL)$. Here the operations $\parallel$ (left-merge) and $|$ (communication) are defined as follows: First we take the case that $X,Y$ consist of finite words only.
The syntax for SEL is by the right-hand side of this definition should be taken with respect to \((\delta,w = \delta)\) and \(\{\delta\} u X = X\), \(x \neq \emptyset\). Informally, \(\text{sym}_2\) 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 \(\text{sym}_2\) does not behave homomorphically). Rather, we introduce an intermediate semantics \(I_1\): 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*)

\[
<c,w> + w.c
\]

We base \(I_1^*\) on \(T_1^*\) just as we based \(I_1\) on \(T_1\). We can now prove

**LEMMA 3.3.** For all \(s, s' \in L_1\) and \(w, w' \in (A \backslash C)^*\)

\[
T_1 \vdash <s, w> + w' \mid <s', w'>
\]

iff

\[
T_1^* \vdash <s, w> + w' \mid <s', w'>
\]

**Proof.** Structural induction on the deductions in \(T_1\) and \(T_1^*\).

This lemma immediately leads to

**THEOREM 3.4.** \(\hat{O}_1[s] = \text{sym}_2(I_1[s])\)

Next we show

**THEOREM 3.5.** \(\hat{I}_1[s] = \hat{D}_1[s]\)

**Proof.** Combine ideas of section 2.4 with a proof that \(I_1\) 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.** \(\hat{O}_1[s] = \text{sym}_2(\hat{O}_1[s])\).

4. THE LANGUAGE \(L_2\): SYNCHRONIZATION MERGE AND GLOBAL NONDETERMINACY

The syntax for \(s \in L_2\) is given by
\[ s := a|c|\text{skip}|\text{fail}|s_1;s_2|s_1+s_2|s_1||s_2|x|ux[s] \]

Here "\( x \)" 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 \( \langle c,w \rangle \Rightarrow \langle \text{fail},w \rangle \). Additionally, we have

(global nondeterminacy)

\[ [u\text{-unfolding}] \]
\[ \langle s_1,w \rangle \Rightarrow \langle s',w \rangle \]
\[ \frac{\langle s_1||s_2,w \rangle \Rightarrow \langle s'_1||s'_2,w \rangle}{\langle s_1+s_2,w \rangle \Rightarrow \langle s',w \rangle} \]

[selection by elementary action]

\[ \frac{\langle s_1,w \rangle \Rightarrow \langle s',w \rangle}{\langle s_1||s_2,w \rangle \Rightarrow \langle s'_1||s'_2,w \rangle} \]

\[ \frac{\langle s_1||s_2,w \rangle \Rightarrow \langle s'_1||s'_2,w \rangle}{\langle s_1+s_2,w \rangle \Rightarrow \langle s',w \rangle} \]

[selection by communication/synchronization]

\[ \frac{\langle s_1||s_2,w \rangle \Rightarrow \langle s'_1||s'_2,w \rangle}{\langle s_1+s_2,w \rangle \Rightarrow \langle s',w \rangle} \]

\[ \frac{\langle s_1||s_2,w \rangle \Rightarrow \langle s'_1||s'_2,w \rangle}{\langle s_1+s_2,w \rangle \Rightarrow \langle s',w \rangle} \]

\[ \langle s_1||s_2,w \rangle \Rightarrow \langle s'_1||s'_2,w \rangle \]

[commutativity of +]

\[ \frac{\langle s_1||s_2,w \rangle \Rightarrow \langle s',w \rangle}{\langle s_1+s_2,w \rangle \Rightarrow \langle s',w \rangle} \]

\[ \frac{\langle s_1||s_2,w \rangle \Rightarrow \langle s',w \rangle}{\langle s_1+s_2,w \rangle \Rightarrow \langle s',w \rangle} \]

\[ \langle s_1+s_2,w \rangle \Rightarrow \langle s',w \rangle \]

\[ \langle s_1||s_2+w \rangle \Rightarrow \langle s',w \rangle \]

\[ \langle s_1+s_2+w \rangle \Rightarrow \langle s',w \rangle \]

\[ \langle s_1||s_2+w \rangle \Rightarrow \langle s',w \rangle \]

\[ \langle s_1+s_2+w \rangle \Rightarrow \langle s',w \rangle \]

Remark. Associativity of + is derivable.

We see that global nondeterminacy is more restrictive than local nondeterminacy. In fact, \( \langle s_1||s_2,w \rangle \Rightarrow \langle s'_1||s'_2,w \rangle \) implies \( \langle s_1+s_2,w \rangle \Rightarrow \langle s'_1+s'_2,w \rangle \) but not vice versa.

Example. \( \langle \text{uc,w},w \rangle \Rightarrow \text{w},\delta \), \( \langle \text{uc,w},w \rangle \Rightarrow \text{w},\delta \), but \( \langle \text{uc,w},w \rangle \Rightarrow \text{w},\delta \) 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 OCM.

4.2. The operational semantics \( \mathcal{O}_2 \).

\( \mathcal{O}_2 \) is derived from \( T_2 \) in the usual way. In addition, however, we now have to consider the case that we have a finite sequence \( \langle s_n,w \rangle \Rightarrow \langle s_{n'},w \rangle \), with no transition \( \langle s_n,w \rangle \Rightarrow \langle s_{n'},w \rangle \), which indeed satisfies \( \mathcal{O}_2 \).

In the BT framework, \( \{[a,0]\} \) is a process for all sets (processes) \( p \), we can do

\[ \mathcal{O}_2 \{[a,0]\} \Rightarrow \{[a,0]\} \]

Example. \( \mathcal{O}_2 \{[a,b]+[a,c]\} \Rightarrow \{[a,b],[b,c]\} \)
without explicitly imposing a counterpart of rule (3.1) for δ.

Operations , ], limits and continuity,
fixed points of contracting operations are as in [3,4,5]. For example, for p,q ∈ P, we put
p || q = (plq) u (q.p) u (p.q) where
plq = \{x: q: x ∈ p\}, all q = \{a,q\}[], q = 1,
\{a,p,q\} q = \{a,p,q\} q, and p|q = U(x<y): x ∈ p,
y ∈ q, where [c,p'][[c,q']] = [{s,p'}][{s,q'}]
,
\{c,p'\}[[c,q']] = [{s,p'}][{s,q'}],
\{c,p'\}[[c,q']] = [{s,p'}][{s,q'}],
c\{c = \{s\}, and (x,y) = ° when x,y are not of one.

of those four forms.

It is now straightforward to define
D_2: I_2 + (I_2 + P), where I_2 = Stmw + P, by
following the clauses in the definition of D_0, D_1.
Thus we put D_2][B] (y) = \{a\}, D_2][s] op s_2^2][B] (y) =
D_2][s_2][o] op D_2][s][o] (y), D_2][s][o] (y) = y(x), and
D_2][s_2][o] (y) = \{x,y\}, where P_0 = {1} and
P_{i+1} = D_2][s][o] (y(x,u))

4. Relationship between D_2 and D_2.

We shall show that
\{a\} D_2^2 = a \cdot D_2.

for suitable a. In fact, a_2^2 is defined in two
steps:
1. First we define syn_2: P + P for p ∈ P

\text{syn}_2(p) = \{a | a ≤ p and a ≤ c\} u
\{a, \text{syn}_2(q) | a ≤ p \} \text{ and a ≤ c\}

For p ∈ P\b_{w}, we have p = \text{lim}_n_{n}^\omega, with P_n ≤ P_n, and we put
\text{syn}_2(p) = \text{lim}_n_{n} (\text{syn}_2(p_n)).

Example. Let p = D_2][s_2][o] (b,c). Then
\text{syn}_2(p) = \{a, b\}, (b, a), b, a, \}, τ).

2. Next, we define traces: P + P(A_2^2) by (finite
case only displayed):

traces (p) = U(traces (x): x ∈ p) if p ≠ °
= \{\} if p = °

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

We now put
a_2^2 = \text{df. traces} \circ \text{syn}_2,

but we cannot (yet) prove (a), because, similarly
\text{to a}_1, a_2^2 does not behave homomorphically.

Therefore, we try an intermediate semantics I_2.
This cannot be based on a simple IR model as the
following argument shows:

Let us try for I_2, similarly to I_1, the addition of
the axiom <c,w> + w.c to T_2. Now consider the
programs s_1 = a; (c_1+c_2), s_2 = (a;c_1) + (a;c_2),
s = c_1. Then D_2[[s_2][u]] = \{at \neq (at,s) =
D_2[[s_2][u]] \}. However, I_2[[s_2][u]] \neq I_2[[s_2][u]] \}. Thus
whatever a we apply to I_2[[s]], the results for
s_1 = s_2 \neq s will turn out the same.

Our solution to this problem is to introduce an
intermediate semantics I_2 which, besides recording
all traces in A_2^2, 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 c from
another parallel compound (for the notion of ready
set cf. [8,11,18,19,21]). Formally, take Δ = P(C).

For X ∈ Δ, let \text{X} = \{X: c ∈ X\}. The ready domain R is
now R = \text{P}(A_2^2 ∪ A_2^2). The transition system T_2
consists of all axioms and rules of T_2 together
with (for \text{w} ∈ A^2).

(i) \text{<c,<w> + w.c}

(ii) \text{<c,<w> + w.(c)}

(iii) \text{<w,ω> + w.ϕ}

(iv) \text{<s_1,<w> + w.X >, <s_2,<w> + w.Y >}

\text{P_{i+1} + s_2,<w> + w.XQY}
For the further results the following lemma is important:

**Lemma 4.1.** For all \( s, s' \in (A \setminus C)^\ast \) the following holds:

1. \( \mathcal{T}_2^* \vdash <s, w> + w' | <s', w'> \) iff \( \mathcal{T}_2^* \vdash <s, w> + w' | <s', w'> \)
2. \( <s, w> \) is a deadlocking configuration for \( \mathcal{T}_2 \) iff there exists some \( X \subseteq C \) with \( \mathcal{T}_2^* \vdash <s, w> + w.X \).

Let now \( w \) range over \( A^\ast \) and let \( W \) range over \( \mathcal{R} (A^\ast \cup A^\ast \Delta) \). We define the abstraction operator \( \text{syn}^\ast : \mathcal{R} \rightarrow A^\ast \) by

\[
\text{syn}_W^\ast (w) = \{ w \mid w \in W \text{ does not contain any } c \in C \} \cup \{ \{ w \} \exists \Delta : w.X = w \}
\]

We have

**THEOREM 4.2.** \( \mathcal{I}_2 = \text{syn}^\ast \mathcal{I}_2 \).

Next, we wish to relate \( \mathcal{I}_2 \) with the full \( \mathcal{R} \) semantics \( \mathcal{D}_2 \). To this end, we introduce the abstraction operator \( \text{readies} : \mathcal{R} \rightarrow \mathcal{D}_2 \) by defining \( \text{readies} (p) \) 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

\[
\text{readies} (p) = \cup \{ \text{readies} (w) : w \in p \} \cup \{ \{ w \} \exists X : w.X \subseteq C \}
\]

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

**THEOREM 4.3.** \( \mathcal{I}_2 = \text{readies} \mathcal{D}_2 \).

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

**LEMMA 4.4.** \( \text{traces} = \text{syn}^\ast \mathcal{I}_2 \) \( \text{readies} \mathcal{D}_2 \)

**THEOREM 4.5.** \( \mathcal{I}_2 = \text{traces} \mathcal{D}_2 \).

Summarizing, we have our final
REFERENCES


