

Syntax and defining equations for an interrupt mechanism in process algebra

Citation for published version (APA):

Baeten, J. C. M., Bergstra, J. A., & Klop, J. W. (1986). Syntax and defining equations for an interrupt mechanism in process algebra. *Fundamenta Informaticae*, 9(2), 127-168.

Document status and date:

Published: 01/01/1986

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

SYNTAX AND DEFINING EQUATIONS FOR AN INTERRUPT MECHANISM IN PROCESS ALGEBRA

J.C.M. BAETEN*, J.A. BERGSTRA** and J.W. KLOP***

* *University of Amsterdam*

** *University of Amsterdam, University of Utrecht*

*** *Centre for Mathematics and Computer Science, Amsterdam*

ABSTRACT

A mechanism is introduced to describe priorities in *ACP*, the algebra of communicating processes, whereby some actions have priority over others in a non-deterministic choice (or sum). This mechanism can be used to model the working of interrupts in a distributed system. This is illustrated in an extensive example.

1980 MATHEMATICS SUBJECT CLASSIFICATION: 68B10, 68C01, 68D25, 68F20.

1982 CR CATEGORIES: F.1.1, F.1.2, F.3.2, F.4.3.

KEY WORDS & PHRASES: concurrency, process algebra, interrupts, priorities.

NOTE: This work was sponsored in part by ESPRIT contract METEOR (nr. 432).

Introduction

Process algebra is an algebraical theory of concurrency, i.e. a theory about concurrent, communicating processes. Almost anything can constitute a process: the execution of a program on a computer, or the execution of an algorithm by a person, but also a game of chess or the behaviour of a vending machine.

The starting point for process algebra is the modular structure of concurrent processes at a given level of abstraction: we consider systems built up from certain basic processes by means of composition tools, including sequencing, alternative choice and parallel composition.

Process algebra tries to find laws or axioms for these composition operators, based on some a priori considerations of what features concurrent communicating processes should certainly have. Thus, we use the axiomatic method; after having established the axioms we can study different models of the theory, thus obtaining actual semantics. We mention some of the ways in which semantics can be obtained:

- * by projective limits as in Bergstra & Klop [4];
- * by topological completion as in de Bakker & Zucker [1];
- * by bisimulations as defined in Park [10] and Milner [9].

In this paper, we will not deal with models at all.

In this introduction, we will just give an informal introduction to the composition operators used in process algebra. For a more technical introduction, we refer the reader to Bergstra & Klop [4], where also a review of related approaches and comparisons with them can be found, or to Bergstra & Tucker [5].

Process algebra starts from a collection of given objects, called atomic actions, atoms or steps. These actions are taken to be indivisible, usually have no duration and form the basic building blocks of our systems. The first two compositional operators we consider are \cdot , denoting sequential composition, and $+$ for alternative composition. If x and y are two processes, then $x \cdot y$ is the process that starts the execution of y after the completion of x , and $x + y$ is the process that chooses either x or y and executes the

chosen process (not the other one). Each time a choice is made, we choose from a set of alternatives. We do not specify whether the choice is made by the process itself, or by the environment. Axioms A1-5 in table 1 below give the laws that $+$ and \cdot obey. We leave out \cdot and brackets as in regular algebra, so $xy + z$ means $(x \cdot y) + z$.

On intuitive grounds $x(y + z)$ and $xy + xz$ present different mechanisms (the moment of choice is different), and therefore, an axiom $x(y + z) = xy + xz$ is not included.

We have a special atom δ denoting deadlock, the acknowledgement of a process that it cannot do anything anymore, the absence of any alternative. Axioms A6-7 give the laws for δ .

Next, we have the parallel composition operator \parallel , called merge. The merge of processes x and y will interleave the actions of x and y , except for the communication actions. In $x \parallel y$, we can either do a step from x , or a step from y , or x and y both synchronously perform an action, which together make up a new action, the communication action. This trichotomy is expressed in axiom CMI. Here, we use two auxiliary operators \sqcup (left-merge) and $|$ (communication merge). Thus, $x \sqcup y$ is $x \parallel y$, but with the restriction that the first step comes from x , and $x|y$ is $x \parallel y$ with a communication step as the first step. Axioms CM2-9 and C1-3 give the laws for \sqcup and $|$. Finally, we have in table 1 the encapsulation operator ∂_H . Here H is a set of atoms, and ∂_H blocks those actions, renames them into ∂ . The operator ∂_H can be used to encapsulate a process, i.e. to block communications with the environment.

Together, these operators make up the axiom system ACP , the algebra of communicating processes, in table 1. In this system, we cannot describe the working of an interrupt. However, for the development of process algebra it is essential to incorporate an interrupt mechanism in order to enhance its expressive power as a specification and verification formalism for concurrent systems.

In this document a new piece of syntax together with semantics defining equations is introduced. Based on a partial ordering on processes, $>$, an operator θ is defined. Now $\theta(x)$ is a context of x inside which action a has priority over b whenever $a > b$. Interrupts will have higher priority than other actions. We think that the operator θ will have many other uses, besides the modelling of interrupts. To the best of our knowledge, the present formalization is new, at least within the theory stream around CCS and CSP.

ACP with the operator θ added will yield the axiom system ACP_θ . In section 2, we prove that this axiom system has nice theoretical properties. In the rest of the paper, we give several examples of the use of θ . For verification purposes, we need an abstraction operator in some of the examples, and therefore we make use of the axiom system ACP_τ , that was introduced in Bergstra & Klop [2]. ACP_τ can be seen as a remodularization of the basic concepts of CCS (see Milner [9]).

We must leave as an open issue whether or not ACP_θ and ACP_τ can be combined into $ACP_{\theta\tau}$. From an intuitive point of view this is not clear as there is no reason why "abstraction" and "interrupts" should commute.

Table of contents:

1. Definitions and motivation
2. Theoretical matters
3. Simple examples
4. Example: a toy distributed system.

1. Definitions and motivation

1.1. Let a, b, c be (atomic) actions, and suppose we want a to take precedence over b and c , we want a to have priority over b and c , which we will express by

$$a > b \quad \text{and} \quad a > c$$

(here $>$ is some partial order on atomic actions). Relative to this partial order, we want to define an operator θ that models this priority, so we want

- i) $\theta(a + b) = a; \theta(a + c) = a;$
- ii) $\theta(b + c) = b + c.$

So θ chooses a over b or c , but θ does not choose between b and c , since they are not $>$ -comparable. We will define θ in an axiomatic way, in the framework of process algebras. Therefore, we will first review the axiom system ACP, the algebra of communicating processes (see Bergstra & Klop [4]).

1.2. The signature of ACP is as follows: A is a given finite set of atoms, $A \subseteq P$, the set of processes. On P we have the following operations:

$+$	alternative composition (sum)
\cdot	sequential composition (product)
\parallel	parallel composition (merge)
$\underline{\parallel}$	left-merge
$ $	communication merge
∂_H	encapsulation
δ	deadlock

The first five operations are binary; ∂_H is a unary operation for each $H \subseteq A$ and $\delta \in A$ is a constant.

1.3. The set of axioms of ACP is as follows (see Bergstra & Klop [4]):

$x + y = y + x$	A1	$a b = b a$	C1
$x + (y + z) = (x + y) + z$	A2	$(a b) c = a (b c)$	C2
$x + x = x$	A3	$\delta a = \delta$	C3
$(x + y)z = xz + yz$	A4		
$(xy)z = x(yz)$	A5	$\partial_H(a) = a$ if $a \notin H$	D1
$x + \delta = x$	A6	$\partial_H(a) = \delta$ if $a \in H$	D2
$\delta x = \delta$	A7	$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3
		$\partial_H(xy) = \partial_H(x) \cdot \partial_H(y)$	D4
$x \parallel y = x \underline{\parallel} y + y \underline{\parallel} x + x y$	CM1		
$a \underline{\parallel} x = ax$	CM2		
$(ax) \underline{\parallel} y = a(x \parallel y)$	CM3		
$(x + y) \underline{\parallel} z = x \underline{\parallel} z + y \underline{\parallel} z$	CM4		
$(ax) b = (a b)x$	CM5		
$a (bx) = (a b)x$	CM6		
$(ax) (by) = (a b)(x \parallel y)$	CM7		
$(x + y) z = x z + y z$	CM8		
$x (y + z) = x y + x z$	CM9		

Table 1.

Here $a, b \in A$, $x, y, z \in P$, $H \subseteq A$.

If we view these equations as rewrite rules, going from left to right, and add a rule A2': $(x + y) + z = x + (y + z)$, we get a term rewrite system RACP. We quote three theorems from Bergstra & Klop [4]:

Theorem. (i) RACP is confluent (has the Church-Rosser property);
(ii) RACP is strongly terminating (working modulo A1 and A2);
(iii) (elimination) for each ACP-term s there is an ACP-term t not containing $\parallel, \sqcup, |, \partial_H$, such that $ACP \vdash s = t$.

1.4 Now we return to the problem in 1.1: we want to extend ACP with an operator θ and give some defining equations for it, so that (i) and (ii) of 1.1 are satisfied. So suppose we have a partial order $<$ on A so that δ is minimal, i.e. we have

1. $\text{not } (a < a)$
2. $a < b \Rightarrow \text{not } (b < a)$
3. $a < b \ \& \ b < c \Rightarrow a < c$
4. $\delta < a$ (if $a \neq \delta$)

for all $a, b, c \in A$.

In order to define θ , we first need to define an auxiliary operator $\triangleleft: P \times P \rightarrow P$.

$$x \triangleleft y$$

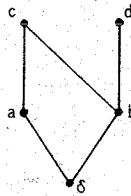
is pronounced x *unless* y . In the following we will explain \triangleleft . First, on A we have

- P1 $a \triangleleft b = a$ if $\text{not } (a < b)$
P2 $a \triangleleft b = \delta$ if $a < b$

So $a \triangleleft b$ is equal to a , *unless* b has priority over a , in which case $a \triangleleft b$ becomes δ . In general, we get $x \triangleleft y$ in the following way:

Suppose x is a sum $\sum_{i=1}^N a_i x_i$, $a_i \in A$, $x_i \in P$. Then, in x we throw away (i.e. set equal to δ) those branches $a_i x_i$ such that there is a branch $b_j y_j$ in y with $a_i < b_j$; other branches we leave untouched.

1.5 Example Suppose $A = \{\delta, a, b, c, d\}$ and we have the following p.o. on A :



(In this picture, we have e.g. $b < c$ because c is located above b and there is a line connecting them; so not $(a < d)$.) We do some calculations: ($x, y, z \in P$).

- a) $(ax + by + c) \triangleleft dz = (ax + by + c) \triangleleft d = ax + \delta + c = ax + c$
- b) $(ax + by + c) \triangleleft c = \delta + \delta + c = c$
- c) $x \triangleleft \delta = x$

1.6 We now present the axiom system ACP_θ (see table 2).

$x + y = y + x$	A1	$a \triangleleft b = a$ if not $(a < b)$	P1
$x + (y + z) = (x + y) + z$	A2	$a \triangleleft b = \delta$ if $a < b$	P2
$x + x = x$	A3	$x \triangleleft yz = x \triangleleft y$	P3
$(x + y)z = xz + yz$	A4	$x \triangleleft (y + z) = (x \triangleleft y) \triangleleft z$	P4
$(xy)z = x(yz)$	A5	$xy \triangleleft z = (x \triangleleft z)y$	P5
$x + \delta = x$	A6	$(x + y) \triangleleft z = x \triangleleft z + y \triangleleft z$	P6
$\delta x = \delta$	A7		
$a b = b a$	C1		
$(a b) c = a (b c)$	C2		
$\delta a = \delta$	C3		
$x \parallel y = x \parallel y + y \parallel x + x y$	CM1	$\theta(a) = a$	TH1
$a \parallel x = ax$	CM2	$\theta(xy) = \theta(x) \cdot \theta(y)$	TH2
$ax \parallel y = a(x \parallel y)$	CM3	$\theta(x + y) = \theta(x) \triangleleft y + \theta(y) \triangleleft x$	TH3
$(x + y) \parallel z = x \parallel z + y \parallel z$	CM4		
$(ax) b = (a b)x$	CM5		
$a (bx) = (a b)x$	CM6		
$(ax) (by) = (a b)(x \parallel y)$	CM7		
$(x + y) z = x z + y z$	CM8		
$x (y + z) = x y + x z$	CM9		
$\partial_H(a) = a$ if $a \notin H$	D1		
$\partial_H(a) = \delta$ if $a \in H$	D2		
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3		
$\partial_H(xy) = \partial_H(x) \cdot \partial_H(y)$	D4		

1.7 Let us verify (i) and (ii) of 1.1 and some other formulas: $(b < a$ and $c < a)$

- i) $\theta(a + b) = \theta(a) \triangleleft b + \theta(b) \triangleleft a = a \triangleleft b + b \triangleleft a = a + \delta = a.$
- ii) $\theta(b + c) = \theta(b) \triangleleft c + \theta(c) \triangleleft b = b \triangleleft c + c \triangleleft b = b + c.$
- iii) $\theta(b(a + c)) = \theta(b) \cdot \theta(a + c) = b(\theta(a) \triangleleft c + \theta(c) \triangleleft a) = b(a \triangleleft c + c \triangleleft a) = b(a + \delta) = ba.$
- iv) $\theta(a + b + c) = \theta(a) \triangleleft (b + c) + \theta(b + c) \triangleleft a = (\theta(a) \triangleleft b) \triangleleft c + (\theta(b) \triangleleft c + \theta(c) \triangleleft b) \triangleleft a = (a \triangleleft b) \triangleleft c + (b \triangleleft c + c \triangleleft b) \triangleleft a = a \triangleleft c + (b + c) \triangleleft a = a + \delta = a.$
- v) $\theta(a + b + c) = \theta(a + b) \triangleleft c + \theta(c) \triangleleft (a + b) = (\theta(a) \triangleleft b + \theta(b) \triangleleft a) \triangleleft c + (\theta(c) \triangleleft a) \triangleleft b = (a \triangleleft b + b \triangleleft a) \triangleleft c + (c \triangleleft a) \triangleleft b = (a + \delta) \triangleleft c + \delta \triangleleft b = a \triangleleft c + \delta \triangleleft c + \delta = a + \delta + \delta = a.$

1.8 In section 2, we will prove that this axiom system is well-behaved. Among other things, we prove a theorem analogous to 1.3 for ACP_θ .

1.9 Note: about leaving out parentheses: we take \cdot to be more binding than other operations and $+$ to be less binding than other operations, so we write $xy \triangleleft z$ for $(xy) \triangleleft z$ and $x + y \triangleleft z$ for $x + (y \triangleleft z)$.

2. Theoretical matters

We will write ACP_θ as a term rewrite system, and prove confluency and termination. We find that ACP_θ is a conservative extension of ACP , and prove an elimination theorem.

2.1 Lemma The following identities hold in ACP_θ :

P7	$(x \triangleleft y) \triangleleft z = (x \triangleleft z) \triangleleft y$
P8	$(x \triangleleft y) \triangleleft y = x \triangleleft y$
TH4	$\theta(x) \triangleleft x = \theta(x)$

Proof. Above the equality signs, we indicate which rule is being used. Let $x, y, z \in P$.

$$P7: (x \triangleleft y) \triangleleft z \stackrel{P4}{=} x \triangleleft (y + z) \stackrel{A1}{=} x \triangleleft (z + y) \stackrel{P4}{=} (x \triangleleft z) \triangleleft y$$

$$P8: (x \triangleleft y) \triangleleft y \stackrel{P4}{=} x \triangleleft (y + y) \stackrel{A3}{=} x \triangleleft y$$

$$TH4: \theta(x) \triangleleft x \stackrel{A3}{=} \theta(x) \triangleleft x + \theta(x) \triangleleft x \stackrel{TH3}{=} \theta(x + x) \stackrel{A3}{=} \theta(x)$$

2.2 Definition: We define the term rewrite system $RACP_\theta$ as follows: take the term rewrite system $RACP$ (see 1.3, rules A1,2,2',3-7,C1-3,CM1-9,D1-4), and add rules $\left\{ \begin{array}{l} P1-8 \\ TH1-4 \end{array} \right\}$
(Reading them as reductions, from left to right)

Note: when proving termination, we will have to work *modulo* rules A1 A2, A2', P7, since applying A1 or P7 twice gives back the original term, and A2' undoes the effect of A2.

We can now state our main theorem:

2.3 Theorem

- i) $RACP_\theta$ is confluent (has the Church-Rosser property);
- ii) $RACP_\theta$ is strongly terminating (working modulo A1, A2, P7);
- iii) (elimination) for each ACP_θ -term s there is a term t not containing $\triangleleft, \theta, ||, |, \partial_H$ such that $ACP_\theta \vdash s = t$.
- iv) ACP_θ is a conservative extension of ACP , i.e. for all ACP -terms s, t we have:

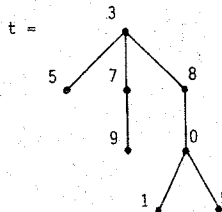
$$ACP \vdash s = t \Leftrightarrow ACP_\theta \vdash s = t$$

Strategy of proof: we first prove (ii) in 2.4-2.12 and then (i) in 2.13-2.15. The elimination (iii) then follows immediately from (ii) and (iv) follows if we combine (i) and (iii).

2.4 To prove (ii), we use the method of *recursive path orderings* of Dershowitz (see Dershowitz [6] and Bergstra & Klop [2]). The idea of this method is, that we assign a certain kind of tree to each term in ACP_θ . We have a partial ordering \Rightarrow on these trees, and it follows from the *Kruskal Tree Theorem* (see Dershowitz [6]) that this ordering is well-founded. Then, if we show that for each reduction in ACP_θ , the trees of the terms before and after the reduction are ordered by \Rightarrow , we have shown that each reduction in ACP_θ must terminate.

2.5 Definition: Let D be the set of all finite commutative rooted trees whose nodes are labeled with natural numbers.

Example:



Notation: $t = 3(5,7(9),8(0,(1,5))) = 3(8(0(1,5)),5,7(9))$.

Definition: We define a partial order \Rightarrow on D as follows: $t = n(t_1, \dots, t_k) \Rightarrow m(s_1, \dots, s_l) = s$ ($k \geq 0, l \geq 0$) iff

- (i) $n > m$ and $t \Rightarrow s_i$ for all $i = 1, \dots, l$; or
- (ii) $n = m$ and for each $i \leq l$ there is a *different* $j \leq k$ such that $t_j \Rightarrow s_i$; also, for at least one $i \leq l$ we have $t_j \Rightarrow s_i$. (so $t_j \neq s_i$); or
- (iii) $n < m$ and $t_i \Rightarrow s$ for some $j \leq k$. \vee

2.6 Theorem (Dershowitz [6]) \Rightarrow is a well-founded partial order on D .

2.7 Let s, t be ACP_θ -terms, and let $s \xrightarrow{r} t$ symbolize that s reduces to t in a one-step, outermost reduction by rule r . Now we want to define a mapping ϕ from terms of ACP_θ to elements of D such that the following hold:

- i) if $s \xrightarrow{r} t$ and $r = A1, A2, A2', P7$, then $\phi(s) = \phi(t)$;
- ii) if $s \xrightarrow{r} t$ and r is any other rule, then $\phi(s) \Rightarrow \phi(t)$. By Dershowitz' theorem, this suffices to show 2.3.ii, the termination of $RACP_\theta$.

2.8 Before we can define ϕ , we need two more definitions.

Definition: Let u be an ACP_θ -term. We define $st(u)$, the *standard part* of u or ACP -part of u , inductively.

- i) $st(u) = u$ if u is an ACP -term;
- ii) if $u \equiv x \square y$, ($\square = +, \cdot, ||, \lfloor, \rfloor$), then $st(u) = st(x) \square st(y)$;
- iii) if $u \equiv \partial_H(x)$, then $st(u) = \partial_H(st(x))$;
- iv) if $u \equiv x \triangleleft y$, then $st(u) = st(x)$;
- v) if $u \equiv \theta(x)$, then $st(u) = st(x)$.

So we obtain $st(u)$ by leaving out all right hand sides y of subterms $x \triangleleft y$, and then leaving out all \triangleleft and θ . It is obvious that $st(u)$ is an ACP -term.

Definition: Let u be an ACP_θ -term. We define $|u|$, the *norm* of u , by

$$|u| = \# \mathcal{G}(st(u)),$$

so $|u|$ is the number of symbols in the reduction graph \mathcal{G} of $st(u)$. Note that an immediate consequence of this definition is that for all ACP_θ -terms x, y :

- a) $|x| + |y| < |x + y|$
- b) $|x| + |y| < |x \cdot y|$.

2.9 Definition: We define ϕ inductively. Let t be an ACP_θ -term

- i) $t \equiv a \in A$. $\phi(a) = .1$ (a single node labeled by 1)
- ii) $t \equiv \partial_H(x)$. Then $\phi(t) =$

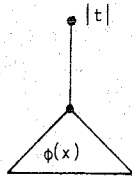


fig.3

iii) $t \equiv x \square y$, with $\square = +, \parallel, \underline{\parallel}$ or $|$

Then $\phi(t) =$

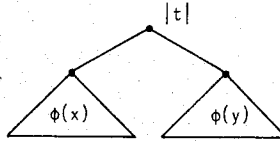


fig.4

iv) $t \equiv x \triangleleft y$. Here we have an intermediate step. First we rewrite x as $(\dots ((z \triangleleft y_1) \triangleleft y_2) \triangleleft \dots \triangleleft y_k)$ ($k \geq 0$), so that the main connective in z is not \triangleleft (in case $k = 0$, we have $z \equiv x$). We use the notation $x = z \triangleleft \{y_1, \dots, y_k\}$, so $t = z \triangleleft \{y, y_1, \dots, y_k\}$, where on the right-hand side of the \triangleleft we have a multiset. Axiom P7 gives us the justification for doing this. Then we define $\phi(t) =$

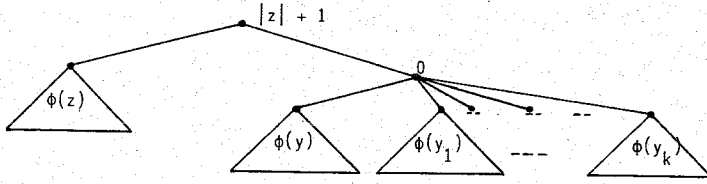


fig.5

If $\sigma = \{y_1, \dots, y_k\}$, we use the abbreviation

$\phi(t) =$

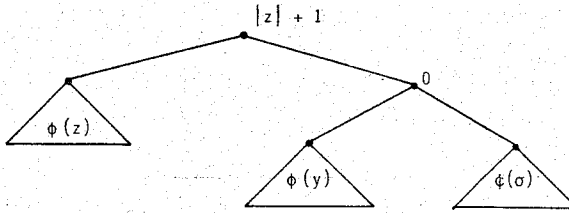


fig.6

v) $t \equiv \theta(x)$. Put $\phi(t) =$

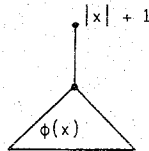


fig.7

$$= \phi(a \triangleleft \sigma).$$

$$\text{P2. } (a < b). \phi(a \triangleleft \{b\} \cup \sigma) =$$

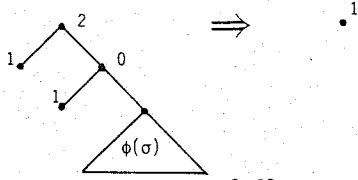


fig.12

$$= \phi(\delta)$$

(we combine this with P8 to get $\delta \triangleleft \sigma = \delta$)

$$\text{P3. } \phi(x \triangleleft \{yz\} \cup \sigma) =$$

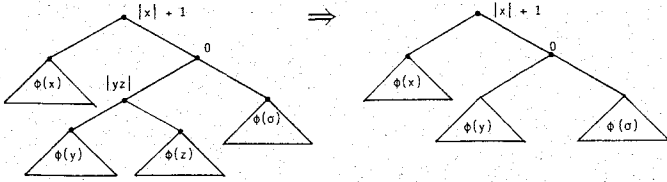


fig.13

$$= \phi(x \triangleleft \{y\} \cup \sigma)$$

$$\text{P4. } \phi(x \triangleleft \{y+z\} \cup \sigma) =$$

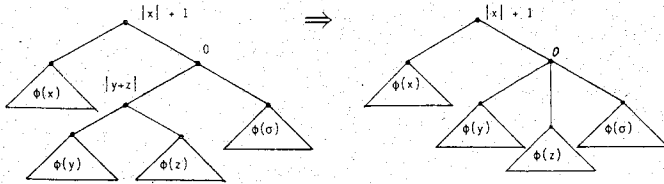


fig.14

$$= \phi(x \triangleleft \{y, z\} \cup \sigma)$$

$$\text{P5. } \phi(xy \triangleleft \sigma) =$$

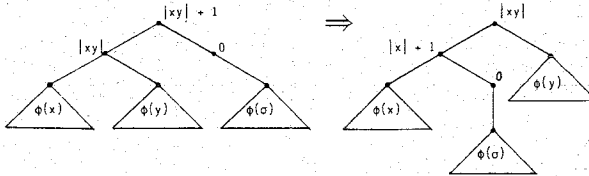


fig.15

$$= \phi((x \triangleleft \sigma)y)$$

(for the proof, use 2.5.i: we have $|xy| + 1 > |xy|$, so we only need that the entire tree on the left majorizes both subtrees on the right)

$$\text{P6. } \phi((x+y) \triangleleft \sigma) =$$

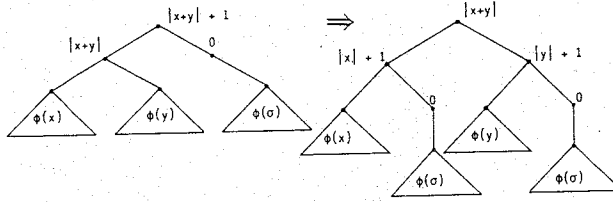


fig.16

$= \phi(x \triangleleft \sigma + y \triangleleft \sigma)$. (again use 2.5.i)

P8. $\phi((x \triangleleft \{y\} \cup \sigma_1) \triangleleft \{y\} \cup \sigma_2) = \phi(x \triangleleft \{y, y\} \cup \sigma) = (\text{write } \sigma = \sigma_1 \cup \sigma_2)$

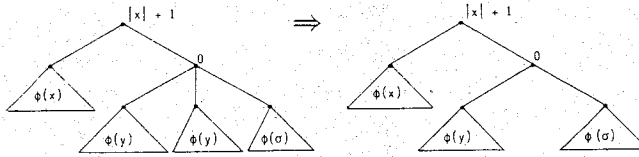


fig.17

$= \phi(x \triangleleft \{y\} \cup \sigma)$ (by 2.5.ii)

TH1. $\phi(\theta(a)) = \downarrow^2 \Rightarrow \cdot 1 = \phi(a)$.

TH2. $\phi(\theta(xy)) = \downarrow^1$

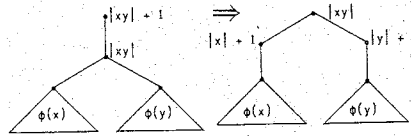


fig.18

$= \phi(\theta(x) \cdot \theta(y))$ (use 2.5.i)

TH3. $\phi(\theta(x+y)) =$

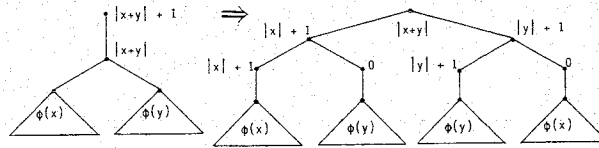


fig.19

$= \phi(\theta(x) \triangleleft y + \theta(y) \triangleleft x)$

(again by 2.5.i). Note that in this last case we should write $x \triangleleft y = z \triangleleft \{y\} \cup \sigma$ and

$y \triangleleft x = w \triangleleft \{x\} \cup \tau$, for some z and w not having \triangleleft as the main connective. But then $|x| \geq |z|$, $|y| \geq |w|$, and the proof of \Rightarrow is not harder.

TH4. $\phi(\theta(x) \triangleleft \{x\} \cup \sigma) =$

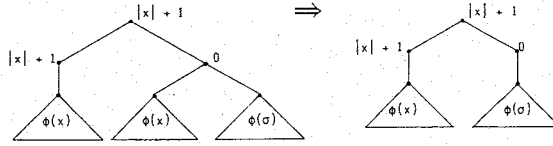


fig.20

$= \phi(\theta(x) \triangleleft \sigma)$ (by 2.5.ii).

2.12 This completes the proof of 2.3.ii: RACP_θ is strongly terminating, working modulo A1, A2, A2', P7.

2.13 Now we turn our attention to proving 2.3.i. Since we know that RACP_θ is strongly terminating, it is enough to show that it is weakly confluent (this follows from Newman's lemma, see Klop [8] or Huet [7]). Showing the weak confluency amounts to looking at all critical pairs (a critical pair is a redex to which two different rules can be applied) and showing that after these reductions a common reduct can be found. In other words: if $t \xrightarrow{r_1} t_1$ and $t \xrightarrow{r_2} t_2$, then there is a term t' such that both t_1 and t_2 reduce to t' (possibly in more than one step). See figure 21.

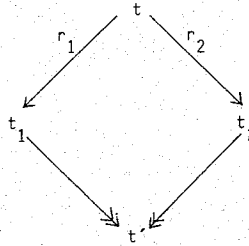


fig.21

We will actually prove more, namely that the *open* theory of RACP_θ is confluent, after adding the extra rules below (by open theory we mean the theory of finite terms, possibly containing variables). The extra rules are:

P9	$x \triangleleft \delta = x$
P10	$\delta \triangleleft x = \delta$
C4	$\delta x = \delta$
C5	$x \delta = \delta$

These rules can be proved very easily for closed finite terms x by induction.

We know already that RACP is confluent (by 1.3), so we only have to check critical pairs involving a new rule (P1-10, TH1-4). Furthermore, C1-5, CM1-9 and D1-4 cannot clash with a new rule.

Still, a tedious job remains. We do the work in 2.14, in cases 2.14.1 to 2.14.46.

2.14 Matrix of critical pairs. * means a proof is required; 0 means redices cannot overlap.

	A1	A2	A3	A4	A5	A6	A7	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	1	2	3	4
P1	0	0	0	0	0	0	0		0	0	0	0	0	0	*	*	*	0	0	0	0
P2	0	0	0	0	0	0	0			0	0	0	0	0	*	*	*	0	0	0	0
P3	0	0	0	*	*	0	*				0	0	0	*	*	0	*	0	0	0	*
P4	*	*	*	0	0	*	0					0	0	*	*	0	*	0	0	0	*
P5	0	0	0	*	*	0	*						0	*	*	*	0	0	0	0	0
P6	*	*	*	0	0	*	0							*	*	0	0	0	0	0	0
P7	0	0	0	0	0	0	0							0	0	0	0	0	0	0	0
P8	0	0	0	0	0	0	0								*	*	*	0	0	0	0
P9	0	0	0	0	0	0	0										0	0	0	0	*
P10	0	0	0	0	0	0	0											0	0	0	0
TH1	0	0	0	0	0	0	0												0	0	*
TH2	0	0	0	*	*	0	*													0	*
TH3	*	*	*	0	0	*	0														*
TH4	0	0	0	0	0	0	0														

1. P1 & P8. Suppose not $(a < b)$

$$(a < b) < b \xrightarrow[P8]{P1} a < b.$$

2. P1 & P9. For all $a \in A$, we have not $(a < \delta)$ (by 1.4.1, 2, 4).

$$a < \delta \xrightarrow[P9]{P1} a.$$

3. P1 & P10. Note that not $(\delta < a)$ implies $a = \delta$. (by 1.4.1, 2, 4)

$$\delta < \delta \xrightarrow[P10]{P1} \delta.$$

4. P2 & P8. Suppose $a < b$.

$$\begin{array}{ccc} (a < b) < b & \xrightarrow{P2} & \delta < b \\ \downarrow P8 & & \downarrow P10 \\ a < b & \xrightarrow{P2} & \delta \end{array}$$

5. P2 & P9. We can never have $a < \delta$.

6. P2 & P10. Let $\delta < a$. $\delta < a \xrightarrow[P10]{P2} \delta$

7. P3 & A4.

$$\begin{array}{ccc} x < (y + z)w & \xrightarrow{P3} & x < (y + z) \xrightarrow{P4} (x < y) < z \\ \downarrow A4 & & \uparrow P3 \\ x < (yw + zw) & \xrightarrow{P4} & (x < yw) < zw \end{array}$$

8. P3 & A5.

$$\begin{array}{ccc}
 x \triangleleft (yz)w & \xrightarrow{P3} & x \triangleleft yz \\
 A5 \downarrow & & \downarrow P3 \\
 x \triangleleft y(zw) & \xrightarrow{P3} & x \triangleleft y
 \end{array}$$

9. P3 & A7. $x \triangleleft \delta y \xrightarrow[A7]{P3} x \triangleleft \delta$.

10. P3 & P7.

$$\begin{array}{ccc}
 (x \triangleleft yz) \triangleleft w & \xrightarrow{P3} & (x \triangleleft y) \triangleleft w \\
 P7 \downarrow & & \downarrow P7 \\
 (x \triangleleft w) \triangleleft yz & \xrightarrow{P3} & (x \triangleleft w) \triangleleft y
 \end{array}$$

11. P3 & P8.

$$\begin{array}{ccc}
 (x \triangleleft yz) \triangleleft yz & \xrightarrow{P3, P3} & (x \triangleleft y) \triangleleft y \\
 P8 \downarrow & & \downarrow P8 \\
 x \triangleleft yz & \xrightarrow{P3} & x \triangleleft y
 \end{array}$$

12. P3 & P10.

$$\begin{array}{ccc}
 \delta \triangleleft xy & \xrightarrow{P3} & \delta \triangleleft x \\
 P10 \downarrow & & \downarrow P10 \\
 \delta & \xleftarrow{} & \delta
 \end{array}$$

13. P3 & TH4.

$$\begin{array}{ccc}
 \theta(xy) \triangleleft xy & \xrightarrow{P3} \theta(xy) \triangleleft x & \xrightarrow{TH2} \theta(x)\theta(y) \triangleleft x \\
 TH4 \downarrow & & \downarrow P5 \\
 \theta(xy) & \xrightarrow{TH2} \theta(x)\theta(y) & \xleftarrow{TH4} (\theta(x) \triangleleft x)\theta(y)
 \end{array}$$

14. P4 & A1.

$$\begin{array}{ccc}
 x \triangleleft (y+z) & \xrightarrow{P4} & (x \triangleleft y) \triangleleft z \\
 A1 \downarrow & & \downarrow P7 \\
 x \triangleleft (z+y) & \xrightarrow{P4} & (x \triangleleft z) \triangleleft y
 \end{array}$$

15. P4 & A2.

$$\begin{array}{ccc}
 x \triangleleft (y+(z+w)) & \xrightarrow{P4} (x \triangleleft y) \triangleleft (z+w) & \xrightarrow{P4} ((x \triangleleft y) \triangleleft z) \triangleleft w \\
 A2 \downarrow & & \uparrow P4 \\
 x \triangleleft ((y+z)+w) & \xrightarrow{P4} (x \triangleleft (y+z)) \triangleleft w &
 \end{array}$$

16. P4 & A3.

$$\begin{array}{ccc}
 x \triangleleft (y+y) & \xrightarrow{P4} & (x \triangleleft y) \triangleleft y \\
 A3 \downarrow & & \downarrow P8 \\
 x \triangleleft y & \xleftarrow{} & x \triangleleft y
 \end{array}$$

17. P4 & A6.

$$\begin{array}{ccc}
 x \triangleleft (y + \delta) & \xrightarrow{P4} & (x \triangleleft y) \triangleleft \delta \\
 \downarrow A6 & & \downarrow P9 \\
 x \triangleleft y & &
 \end{array}$$

18. P4 & P7.

$$\begin{array}{ccc}
 (x \triangleleft (y + z)) \triangleleft w & \xrightarrow{P4} & ((x \triangleleft y) \triangleleft z) \triangleleft w \\
 \downarrow P7 & & \downarrow P7, P7 \\
 (x \triangleleft w) \triangleleft (y + z) & \xrightarrow{P4} & ((x \triangleleft w) \triangleleft y) \triangleleft z
 \end{array}$$

19. P4 & P8.

$$\begin{array}{ccc}
 (x \triangleleft (y + z)) \triangleleft (y + z) & \xrightarrow{P4, P4} & (((x \triangleleft y) \triangleleft z) \triangleleft y) \triangleleft z \\
 \downarrow P8 & & \downarrow P7 \\
 x \triangleleft (y + z) & \xrightarrow{P4} & (x \triangleleft y) \triangleleft z \xleftarrow{P8, P8} (((x \triangleleft y) \triangleleft y) \triangleleft z) \triangleleft z
 \end{array}$$

20. P4 & P10.

$$\begin{array}{ccc}
 \delta \triangleleft (x + y) & \xrightarrow{P4} & (\delta \triangleleft x) \triangleleft y \\
 \downarrow P10 & & \downarrow P10 \\
 \delta & \xleftarrow{P10} & \delta \triangleleft y
 \end{array}$$

21. P4 & TH4.

$$\begin{array}{ccc}
 \theta(x + y) \triangleleft (x + y) & \xrightarrow{P4} & (\theta(x + y) \triangleleft x) \triangleleft y \xrightarrow{TH3} ((\theta(x) \triangleleft y + \theta(y) \triangleleft x) \triangleleft x) \triangleleft y \\
 \downarrow TH4 & & \downarrow P6, P7 \\
 \theta(x + y) & \xrightarrow{TH3} & \theta(x) \triangleleft y + \theta(y) \triangleleft x \xleftarrow{P8, TH2} ((\theta(x) \triangleleft x) \triangleleft y) \triangleleft y + ((\theta(y) \triangleleft y) \triangleleft x) \triangleleft x
 \end{array}$$

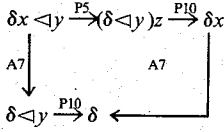
22. P5 & A4.

$$\begin{array}{ccc}
 (x + y)z \triangleleft w & \xrightarrow{P5} & ((x + y) \triangleleft w)z \xrightarrow{P6} (x \triangleleft w + y \triangleleft w)z \\
 \downarrow A4 & & \downarrow A4 \\
 (xz + yz) \triangleleft w & \xrightarrow{P6} & xz \triangleleft w + yz \triangleleft w \xrightarrow{P5} (x \triangleleft w)z + (y \triangleleft w)z
 \end{array}$$

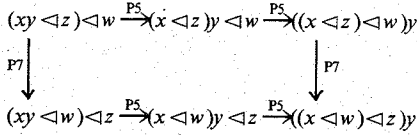
23. P5 & A5.

$$\begin{array}{ccc}
 (xy)z \triangleleft w & \xrightarrow{P5} & (xy \triangleleft w)z \\
 \downarrow A5 & & \downarrow P5, A5 \\
 x(yz) \triangleleft w & \xrightarrow{P5} & (x \triangleleft w)yz
 \end{array}$$

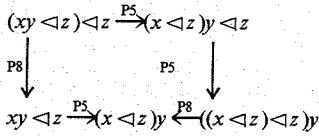
24. P5 & A7.



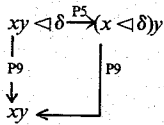
25. P5 & P7.



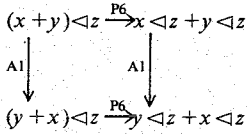
26. P5 & P8.



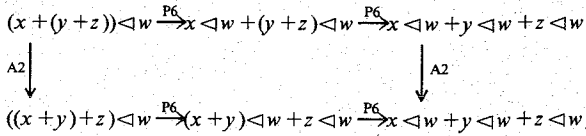
27. P5 & P9.



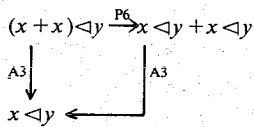
28. P6 & A1.



29. P6 & A2.



30. P6 & A3.



31. P6 & A6.

$$\begin{array}{ccc}
 (x + \delta) \triangleleft y & \xrightarrow{P6} & x \triangleleft y + \delta \triangleleft y \\
 \downarrow A6 & & \downarrow P10 \\
 x \triangleleft y & \xleftarrow{A6} & x \triangleleft y + \delta
 \end{array}$$

32. P6 & P7.

$$\begin{array}{ccccc}
 ((x + y) \triangleleft z) \triangleleft w & \xrightarrow{P6} & (x \triangleleft z + y \triangleleft z) \triangleleft w & \xrightarrow{P6} & (x \triangleleft z) \triangleleft w + (y \triangleleft z) \triangleleft w \\
 \downarrow P7 & & & & \downarrow P7 \\
 ((x + y) \triangleleft w) \triangleleft z & \xrightarrow{P6} & (x \triangleleft w + y \triangleleft w) \triangleleft z & \xrightarrow{P6} & (x \triangleleft w) \triangleleft z + (y \triangleleft w) \triangleleft z
 \end{array}$$

33. P6 & P8.

$$\begin{array}{ccc}
 ((x + y) \triangleleft z) \triangleleft z & \xrightarrow{P6} & (x \triangleleft z + y \triangleleft z) \triangleleft z \\
 \downarrow P8 & & \downarrow P6 \\
 (x + y) \triangleleft z & \xrightarrow{P6} & x \triangleleft z + y \triangleleft z \xleftarrow{P8, P8} (x \triangleleft z) \triangleleft z + (y \triangleleft z) \triangleleft z
 \end{array}$$

34. P8 & P9.

$$(x \triangleleft \delta) \triangleleft \delta \xrightarrow{P8, P9} x \triangleleft \delta$$

35. P8 & P10.

$$(\delta \triangleleft x) \triangleleft x \xrightarrow{P8, P10} \delta \triangleleft x$$

36. P9 & TH4.

$$\theta(\delta) \triangleleft \delta \xrightarrow{P9, TH4} \theta(\delta)$$

37. TH1 & TH4.

$$\begin{array}{ccc}
 \theta(a) \triangleleft a & \xrightarrow{TH1} & a \triangleleft a \\
 \downarrow TH4 & & \downarrow P1, \text{ use 1.4.1} \\
 \theta(a) & \xrightarrow{TH1} & a
 \end{array}$$

38. TH2 & A4

$$\begin{array}{ccccccc}
 \theta((x + y)z) & \xrightarrow{TH2} & \theta(x + y)\theta(z) & \xrightarrow{TH3} & (\theta(x) \triangleleft y + \theta(y) \triangleleft x)\theta(z) & \xrightarrow{A4} & \theta(z) \triangleleft y)\theta(z) + (\theta(y) \triangleleft x)\theta(z) \\
 \downarrow A4 & & & & & & \uparrow P5 \\
 \theta(xz + yz) & \xrightarrow{TH3} & \theta(xz) \triangleleft yz + \theta(yz) \triangleleft xz & \xrightarrow{P3, TH2} & \theta(x)\theta(z) \triangleleft y + \theta(y)\theta(z) \triangleleft x & &
 \end{array}$$

39. TH2 & A5.

$$\begin{array}{ccc}
 \theta((xy)z) & \xrightarrow{TH2} & \theta(xy)\theta(z) \xrightarrow{TH2} \theta(x)\theta(y)\theta(z) \\
 \downarrow A5 & & \downarrow A5 \\
 \theta(x(yz)) & \xrightarrow{TH2} & \theta(x)\theta(yz) \xrightarrow{TH2} \theta(x)\theta(y)\theta(z)
 \end{array}$$

40. TH2 & A7.

$$\begin{array}{ccc}
 \theta(\delta x) & \xrightarrow{\text{TH2}} & \theta(\delta)\theta(x) \\
 \text{A7} \downarrow & & \text{TH1} \downarrow \\
 \theta(\delta) & \xrightarrow{\text{TH1}} & \delta \xleftarrow{\text{A7}} \delta\theta(x)
 \end{array}$$

41. TH2 & TH4.

$$\begin{array}{ccc}
 \theta(xy) \triangleleft xy & \xrightarrow{\text{TH2}} & \theta(x)\theta(y) \triangleleft xy \xrightarrow{\text{P3}} \theta(x)\theta(y) \triangleleft x \\
 \text{TH4} \downarrow & & \text{P5} \downarrow \\
 \theta(xy) & \xrightarrow{\text{TH2}} & \theta(x)\theta(y) \xleftarrow{\text{TH4}} (\theta(x) \triangleleft x)\theta(y)
 \end{array}$$

42. TH3 & A1.

$$\begin{array}{ccc}
 \theta(x+y) & \xrightarrow{\text{TH3}} & \theta(x) \triangleleft y + \theta(y) \triangleleft x \\
 \text{A1} \downarrow & & \downarrow \text{A1} \\
 \theta(y+x) & \xrightarrow{\text{TH3}} & \theta(y) \triangleleft x + \theta(x) \triangleleft y
 \end{array}$$

43. TH3 & A2.

$$\begin{array}{ccc}
 \theta(x + (y + z)) & \xrightarrow{\text{TH3}} & \theta(x) \triangleleft (y + z) + \theta(y + z) \triangleleft x \xrightarrow{\text{TH3,P4}} \\
 \text{A2} \downarrow & & \downarrow \\
 \theta((x + y) + z) & & (\theta(x) \triangleleft y) \triangleleft z + (\theta(y) \triangleleft z + \theta(z) \triangleleft y) \triangleleft x \\
 \text{TH3} \downarrow & & \downarrow \text{P6} \\
 \theta(x + y) \triangleleft z + \theta(z) \triangleleft (x + y) & & (\theta(x) \triangleleft y) \triangleleft z + (\theta(y) \triangleleft z) \triangleleft x + (\theta(z) \triangleleft y) \triangleleft x \\
 \text{TH3,P4} \downarrow & & \downarrow \text{P7} \\
 (\theta(x) \triangleleft y + \theta(y) \triangleleft x) \triangleleft z + (\theta(z) \triangleleft x) \triangleleft y & & \\
 \text{P6} \downarrow & & \\
 (\theta(x) \triangleleft y) \triangleleft z + (\theta(y) \triangleleft x) \triangleleft z + (\theta(z) \triangleleft y) \triangleleft x
 \end{array}$$

44. TH3 & A3.

$$\begin{array}{ccc}
 \theta(x+x) & \xrightarrow{\text{TH3}} & \theta(x) \triangleleft x + \theta(x) \triangleleft x \\
 \text{A3} \downarrow & & \downarrow \text{A3} \\
 \theta(x) & \xleftarrow{\text{TH4}} & \theta(x) \triangleleft x \xleftarrow{\text{A3}}
 \end{array}$$

45. TH3 & A6.

$$\begin{array}{ccc}
 \theta(x + \delta) & \xrightarrow{\text{TH3}} & \theta(x) \triangleleft \delta + \theta(\delta) \triangleleft x \\
 \text{A6} \downarrow & & \downarrow \text{P9,TH1} \\
 \theta(x) & \xleftarrow{\text{A6}} & \theta(x) + \delta \xleftarrow{\text{P10}} \theta(x) + \delta \triangleleft x
 \end{array}$$

46. TH3 & TH4.

$$\begin{array}{c}
\theta(x+y) \triangleleft (x+y) \xrightarrow{\text{TH3}} (\theta(x) \triangleleft y + \theta(y) \triangleleft x) \triangleleft (x+y) \xrightarrow{\text{P4,P6}} \\
\text{TH4} \downarrow ((\theta(x) \triangleleft y) \triangleleft x) \triangleleft y + ((\theta(y) \triangleleft x) \triangleleft y) \triangleleft y \xleftarrow{\text{P7,P7}} \\
\theta(x+y) \xrightarrow{\text{TH3}} ((\theta(x) \triangleleft x) \triangleleft y) \triangleleft y + ((\theta(y) \triangleleft y) \triangleleft x) \triangleleft x \\
\theta(x) \triangleleft y + \theta(y) \triangleleft x \xrightarrow{\text{TH4,TH4}} \theta(x) \triangleleft y + \theta(y) \triangleleft x \triangleleft x \\
\text{P8,P8} \uparrow (\theta(x) \triangleleft y) \triangleleft y + (\theta(y) \triangleleft x) \triangleleft x
\end{array}$$

2.15 This completes the proof of 2.3.i, and thereby the proof of 2.3.

2.16 **Theorem** The following identities hold in the initial term model A_ω :

- i) either $a \triangleleft x = a$ or $a \triangleleft x = \delta$
- ii) $a \triangleleft (x \triangleleft b) = a \triangleleft x$ if not $(a < b)$ (P11)
- iii) $(x \triangleleft b) \triangleleft a = x \triangleleft b$ if $a < b$ (P12)
- iv) $(x \triangleleft y) \triangleleft (z \triangleleft y) = (x \triangleleft y) \triangleleft z$ (P13)
- v) $\theta(x \triangleleft y) = \theta(x) \triangleleft y$ (TH5)

Proof All proofs are by induction on terms. An induction on x needs to consider only three cases:

1) $x \equiv a$, an atom; (2) $x \equiv ay$; (3) $x \equiv y + z$ (this uses 2.3.iii, we can eliminate $\triangleleft, \theta, \parallel, \ll, |, \partial_H$).

i) Induction on x

case 1: suppose $x \equiv b$

case 1.1 not $(a < b)$. Then $a \triangleleft x = a \triangleleft b = a$

case 1.2 $a < b$. Then $a \triangleleft x = a \triangleleft b = \delta$

case 2: $x \equiv yz$ and suppose (i) holds for y . We have $a \triangleleft x = a \triangleleft yz = a \triangleleft y$. Now apply the induction hypothesis.

case 3: $x \equiv y + z$ and suppose (i) holds for y and z .

case 3.1 $a \triangleleft y = a$. Then $a \triangleleft x = a \triangleleft (y + z) = (a \triangleleft y) \triangleleft z = a \triangleleft z$. Now apply induction hypothesis.

case 3.2 $a \triangleleft y = \delta$. Then $a \triangleleft x = (a \triangleleft y) \triangleleft z = \delta \triangleleft z = \delta$.

ii) induction on x . Suppose not $(a < b)$.

case 1: $x \equiv c$

case 1.1 not $(c < b)$. Then $a \triangleleft (c \triangleleft b) = a \triangleleft c$.

case 1.2 $c < b$. Note that then we must have not $(a < c)$. So $a \triangleleft (c \triangleleft b) = a \triangleleft \delta = a = a \triangleleft c$.

case 2: $x \equiv cy$. $a \triangleleft (cy \triangleleft b) = a \triangleleft (c \triangleleft b)y = a \triangleleft (c \triangleleft b) = a \triangleleft c = a \triangleleft cy$.

case 3: $x \equiv y + z$, and suppose (ii) holds for y and z . $a \triangleleft ((y + z) \triangleleft b) = a \triangleleft (y \triangleleft b + z \triangleleft b) = (a \triangleleft (y \triangleleft b)) \triangleleft (z \triangleleft b) = (a \triangleleft y) \triangleleft (z \triangleleft b) = (a \triangleleft (z \triangleleft b)) \triangleleft y = (a \triangleleft z) \triangleleft y = a \triangleleft (y + z)$.

iii) induction on x . Suppose $a < b$.

case 1: $x \equiv c$.

case 1.1 Not $(c < b)$. Note that then we must have not $(c < a)$. So $(c \triangleleft b) \triangleleft a = c \triangleleft a = c = c \triangleleft b$.

case 1.2 $c < b$. $(c \triangleleft b) \triangleleft a = \delta \triangleleft a = \delta = c \triangleleft b$.

case 2: $x \equiv cy$. $(cy \triangleleft b) \triangleleft a = (c \triangleleft b)y \triangleleft a = ((c \triangleleft b) \triangleleft a)y = (c \triangleleft b)y = cy \triangleleft b$

case 3: $x \equiv y + z$, and suppose (iii) holds for y and z . $((y + z) \triangleleft b) \triangleleft a = (y \triangleleft b + z \triangleleft b) \triangleleft a = (y \triangleleft b) \triangleleft a + (z \triangleleft b) \triangleleft a = y \triangleleft b + z \triangleleft b = (y + z) \triangleleft b$.

iv) Induction on z .

case 1: $z \equiv a$.

case 1.1 $a \triangleleft y = a$. Then $(x \triangleleft y) \triangleleft (a \triangleleft y) = (x \triangleleft y) \triangleleft a$.

case 1.2 $a \triangleleft y = \delta$. In this case, we will prove the following variant of (iii) by induction on y :

(*) if $a \triangleleft y = \delta$, then $(x \triangleleft y) \triangleleft a = x \triangleleft y$.

case 1.2.1 $y \equiv b$. This is (iii).

case 1.2.2 $y \equiv bw$. By 1.2.1 we have $(x \triangleleft bw) \triangleleft a = (x \triangleleft b) \triangleleft a = x \triangleleft b = x \triangleleft bw$, for $\delta = a \triangleleft y = a \triangleleft b$.

case 1.2.3 $y \equiv w + v$. We have $\delta = a \triangleleft y = (a \triangleleft v) \triangleleft w$. We know by (i) that either $a \triangleleft v = a$ or $a \triangleleft v = \delta$. If $a \triangleleft v = a$, we have $a \triangleleft w = \delta$. Thus we have either $a \triangleleft v = \delta$ or $a \triangleleft w = \delta$. Say $a \triangleleft v = \delta$ (the proof is very similar in the other case).

Then $(x \triangleleft y) \triangleleft a = (x \triangleleft (v + w)) \triangleleft a = ((x \triangleleft v) \triangleleft w) \triangleleft a = ((x \triangleleft v) \triangleleft a) \triangleleft w = (x \triangleleft v) \triangleleft w = x \triangleleft (v + w)$.

Thus we have proved (*), and now $(x \triangleleft y) \triangleleft (a \triangleleft y) = (x \triangleleft y) \triangleleft \delta = x \triangleleft y = (x \triangleleft y) \triangleleft a$ follows.

case 2: $z \equiv aw$. By case 1, $(x \triangleleft y) \triangleleft (aw \triangleleft y) = (x \triangleleft y) \triangleleft (a \triangleleft y)w = (x \triangleleft y) \triangleleft (a \triangleleft y) = (x \triangleleft y) \triangleleft a = (x \triangleleft y) \triangleleft aw$.

case 3: $z \equiv w + v$, and suppose (iv) holds for w and v . Then $(x \triangleleft y) \triangleleft ((v + w) \triangleleft y) = (x \triangleleft y) \triangleleft (v \triangleleft y + w \triangleleft y) = ((x \triangleleft y) \triangleleft (v \triangleleft y)) \triangleleft (w \triangleleft y) = ((x \triangleleft y) \triangleleft v) \triangleleft (w \triangleleft y) = ((x \triangleleft y) \triangleleft (w \triangleleft y)) \triangleleft v = ((x \triangleleft y) \triangleleft w) \triangleleft v = (x \triangleleft y) \triangleleft (w + v)$.

v) Induction on x .

case 1: $x \equiv a$.

case 1.1 $a \triangleleft y = a$. Then $\theta(a \triangleleft y) = \theta(a) = a = a \triangleleft y = \theta(a) \triangleleft y$.

case 1.2 $a \triangleleft y = \delta$. Then $\theta(a \triangleleft y) = \theta(\delta) = \delta = a \triangleleft y = \theta(a) \triangleleft y$.

case 2: $x \equiv az$. By case 1, $\theta(az \triangleleft y) = \theta((a \triangleleft y)z) = \theta(a \triangleleft y)\theta(z) = (\theta(a) \triangleleft y)\theta(z) = \theta(a)\theta(z) \triangleleft y = \theta(az) \triangleleft y$.

case 3: $x \equiv z + w$ and suppose (v) holds for z and w . Then $\theta((z + w) \triangleleft y) = \theta(z \triangleleft y + w \triangleleft y) = \theta(z \triangleleft y) \triangleleft (w \triangleleft y) + \theta(w \triangleleft y) \triangleleft (z \triangleleft y) = (\theta(z) \triangleleft y) \triangleleft (w \triangleleft y) + (\theta(w) \triangleleft y) \triangleleft (z \triangleleft y) = (\text{apply (iv)!}) = (\theta(z) \triangleleft y) \triangleleft w + (\theta(w) \triangleleft y) \triangleleft z = (\theta(z) \triangleleft w) \triangleleft y + (\theta(w) \triangleleft z) \triangleleft y = (\theta(z) \triangleleft w + \theta(w) \triangleleft z) \triangleleft y = \theta(z + w) \triangleleft y$.

2.17 Note: Adding equations P11-13 and TH5 to RACP_θ as rewrite rules (reading from left to right) also gives a terminating and confluent rewrite system. The many and tedious details of the proof of this claim we happily leave to the reader.

3. Simple examples

We will now give some simple examples that use priorities, as defined in ACP_θ , and give some motivation of the choice of priorities.

We distinguish three cases:

1. give priority to interrupts;
2. give lower priority to time outs, error messages;
3. give priority to internal actions, real time behaviour.

3.1 Example 1:

Let D be a finite set of data, and suppose we have an infinite sequence of data from D ,

$$\alpha = \langle d_0, d_1, d_2, \dots \rangle$$

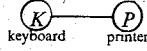


fig.22

The printer will print these data, but can be interrupted by the keyboard. We have the following atomic actions A : (a) actions of K :

1. $k(BR)$ = key in BREAK
2. $k(SP)$ = key in START PRINTING
3. $s(BR)$ = send BR to the printer
4. $s(SP)$ = send SP to the printer

recursive equation for K :

$$K = [k(BR)s(BR) + k(SP)s(SP)]K.$$

(b) actions of P :

1. $p(d)$ = print symbol d ;
2. $r(BR)$ = receive BR from the key-board;
3. $r(SP)$ = receive SP from the key-board.

The printer has two states, Printing and Waiting. We let P_i stand for printing state after having printed d_0, d_1, \dots, d_{i-1} ; and W_i stand for waiting state after having printed d_0, d_1, \dots, d_{i-1} ($i \geq 0$).

Equations for P :

$$\begin{aligned} P &= W_0 \\ W_i &= r(SP)P_i + r(BR)W_i \\ P_i &= p(d_i)P_{i+1} + r(BR)W_i + r(SP)P_i \end{aligned}$$

3.2 Now we define the communication function by:

$$\begin{aligned} s(BR)|r(BR) &= br \\ s(SP)|r(SP) &= sp \end{aligned}$$

and all other communications give δ . We are interested in

$$K \parallel P$$

but want to hide unsuccessful communications. Therefore, we define

$$H = \{s(BR), r(BR), s(SP), r(SP)\}$$

and look at

$$\partial_H(K \parallel P)$$

Now we want to define a partial order on atomic actions giving priority. Note that both br and sp should have priority over printing actions (br must interrupt printing and $s(SP)$ must be received by the printer, otherwise the key-board will be blocked and a break cannot be given). All other priorities are given by 1.4.4. We have the following picture of the partial order (figure 23).

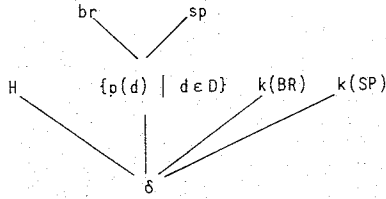


fig.23

Here if two actions a and b are connected by a line, and b is above a , then $a < b$. If two actions are not connected, they are incomparable. If we put a set at a certain position, it means each element of the set has that position. If θ is defined using this partial order, we can describe the system by:

$$\boxed{\theta \circ \partial_H (K \parallel P)}$$

3.3 Theorem. Put $\mathcal{P}_i = \theta \circ \partial_H (K \parallel P_i)$ and $\mathcal{W}_i = \theta \circ \partial_H (K \parallel W_i)$ ($i \geq 0$), then we have the following equations:

1. $\mathcal{W}_i = k(BR) \cdot br \cdot \mathcal{W}_i + k(SP) \cdot sp \cdot \mathcal{P}_i.$
2. $\mathcal{P}_i = p(d_i) \mathcal{P}_{i+1} + k(BR) \cdot br \cdot \mathcal{W}_i + k(SP) \cdot sp \cdot \mathcal{P}_i. \quad (i \geq 0).$

Proof: as we go along, we will skip some steps. To calculate the merge, we need the *expansion theorem* (see Bergstra & Tucker [5]) which goes as follows:

if X_1, \dots, X_k are given, put \mathbf{X}^i = the merge of all X_n except X_i and $\mathbf{X}^{i,j}$ = the merge of all X_n except X_i and X_j . Then

$$(ET) X_i \parallel \dots \parallel X_k = \sum_{1 \leq i \leq k} X_i \parallel \mathbf{X}^i + \sum_{1 \leq i < j \leq k} (X_i \mid X_j) \parallel \mathbf{X}^{i,j},$$

i.e. we can start with an action of one of the processes, or with a communication between two of them.

Let $i \geq 0$.

$$\begin{aligned}
 \mathcal{W}_i &= \theta \circ \partial_H (K \parallel W_i) = & (1) \\
 &= \theta \circ \partial_H (k(BR)(s(BR)K \parallel W_i) + \\
 &\quad + k(SP)(s(SP)K \parallel W_i) + \\
 &\quad + r(BR)(K \parallel W_i) + \\
 &\quad + r(SP)(K \parallel P_i)) = \\
 &= \theta(k(BR) \cdot \partial_H (s(BR)K \parallel W_i) + \\
 &\quad + k(SP) \cdot \partial_H (s(SP)K \parallel W_i)) = \\
 &= k(BR) \cdot \theta \circ \partial_H (s(BR)(K \parallel W_i) + r(SP)(s(BR)K \parallel P_i) + \\
 &\quad + r(BR)(s(BR)K \parallel W_i) + \\
 &\quad + (s(BR) \mid r(BR))(K \parallel W_i)) + \\
 &\quad + k(SP) \cdot \theta \circ \partial_H (s(SP)(K \parallel W_i) + r(SP)(s(SP)K \parallel P_i) + \\
 &\quad + r(BR)(s(SP)K \parallel W_i) + \\
 &\quad + (s(SP) \mid r(SP))(K \parallel P_i)) =
 \end{aligned}$$

$$\begin{aligned}
&= k(BR)\theta(br.\partial_H(K\|W_i)) + k(SP)\theta(sp.\partial_H(K\|P_i)) = \\
&= k(BR)br.\mathcal{W}_i + k(SP)sp.\mathcal{P}_i.
\end{aligned}$$

$$\begin{aligned}
\mathcal{P}_i &= \theta.\partial_H(K\|P_i) = \\
&= \theta(k(BR)br.\partial_H(s(BR)K\|P_i) + \\
&\quad + k(SP)sp.\partial_H(s(SP)K\|P_i) + \\
&\quad + p(d_i)br.\partial_H(K\|P_{i+1})) = \\
&= k(BR)\theta(p(d_i)br.\partial_H(s(BR)K\|P_{i+1}) + \\
&\quad + (s(BR)|r(BR))\partial_H(K\|W_i)) + \\
&\quad + k(SP)\theta(p(d_i)sp.\partial_H(s(SP)K\|P_{i+1}) + \\
&\quad + (s(SP)|r(SP))\partial_H(K\|P_i)) + \\
&\quad + p(d_i)\mathcal{P}_{i+1} = \\
&= (\text{this is where we use the priority}) \\
&\quad k(BR)br.\mathcal{W}_i + k(SP)sp.\mathcal{P}_i + p(d_i)\mathcal{P}_{i+1}.
\end{aligned} \tag{2}$$

We can make the following state transition diagram:

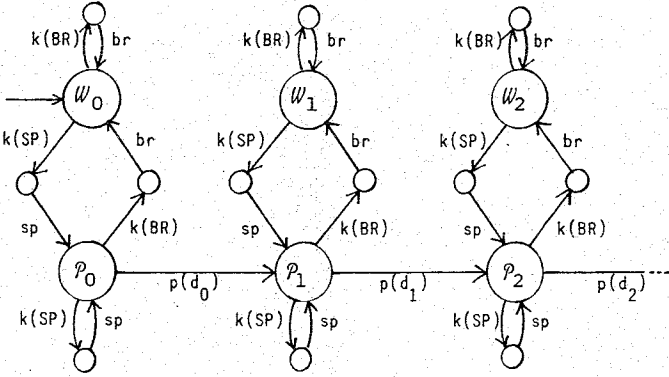


fig.24

3.4 Example 2: Suppose we have a file F containing an infinite sequence of data from a finite set D , so F contains

$$\alpha = \langle d_0, d_1, \dots \rangle.$$

These data can be sent to the printer and subsequently printed, or a file crash might occur, in which case an error statement will be generated.

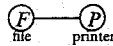


fig.25

We have the following atomic actions A :

(a) actions of F :

1. $g(d) = \text{get the next symbol, } d \in D, \text{ from the file}$
2. $s(d) = \text{send } d \in D \text{ to the printer}$
3. $cr = \text{file crash.}$

Let F_i stand for the state of F after d_0, \dots, d_{i-1} have been sent ($i \geq 0$). Then F is described by the following equations:

$$\begin{aligned} F &= F_0 \\ F_i &= g(d_i) \cdot s(d_i) \cdot F_{i+1} + cr \quad (i \geq 0). \end{aligned}$$

(b) actions of P :

1. $r(d) = \text{receive } d \in D \text{ from the file}$
2. $p(d) = \text{print } d \in D$
3. $o(CR) = \text{observe file crash}$
4. $p(CR) = \text{print 'FILE HAS CRASHED'}$.

Recursive equation for P :

$$P = \sum_{d \in D} r(d) \cdot p(d) \cdot P + o(CR) \cdot p(CR).$$

(c) communication actions:

if $d \in D$, then $r(d) \mid s(d) = c(d)$ (*communicate* d). All other communications give δ . Since we want to hide all unsuccessful communications, we define $H = \{r(d), s(d) \mid d \in D\}$, and look at

$$\partial_H(F \parallel P).$$

3.5 Priority. Here the rationale of defining priorities is different. The file-crash cr might occur at any moment, but $o(CR)$, the observation of a file-crash, can only occur if the file has actually crashed. We ensure this by giving $o(CR)$ a lower priority than every 'regular' action of F , so we must have $o(CR) < g(d)$ and $o(CR) < c(d)$. This gives the following picture (figure 26).

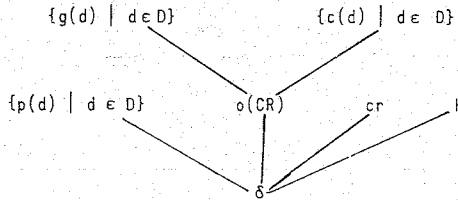


fig.26

If θ is defined using this partial order, we can describe the system by:

$$\theta \circ \partial_H(F \parallel P)$$

3.6 Theorem: Put $\mathcal{F}_i = \theta \circ \partial_H(F_i \parallel P)$, and $\mathcal{P}_i = \theta \circ \partial_H(F_{i+1} \parallel p(d_i)P)$ ($i \geq 0$), then we have the following equations:

- 1) $\mathfrak{F}_i = g(d_i) \cdot c(d_i) \cdot \mathfrak{P}_i + cr \cdot o(CR) \cdot p(CR)$
- 2) $\mathfrak{P}_i = g(d_{i+1}) \cdot p(d_i) \cdot c(d_{i+1}) \cdot P_{i+1} +$
 $+ cr \cdot p(d_i) \cdot o(CR) \cdot p(CR) + p(d_i) \cdot \mathfrak{F}_{i+1} \ (i \geq 0)$

Proof let $i \geq 0$.

$$\begin{aligned}
 \mathfrak{F}_i &= \theta \circ \partial_H(F_i \| P) & (1) \\
 &= \theta(g(d_i) \cdot \partial_H(s(d_i)F_{i+1} \| P) + cr \cdot \partial_H(P) + \\
 &\quad + o(CR) \cdot \partial_H(F_i \| p(CR))) = \\
 &= (\text{use } g(d_i) > o(CR)) \\
 &\quad g(d_i) \cdot \theta((s(d_i) \mid r(d_i)) \cdot \partial_H(F_{i+1} \| p(d_i)P) \\
 &\quad + o(CR) \cdot \partial_H(s(d_i)F_{i+1} \| p(CR))) + \\
 &\quad + cr \cdot \theta(o(CR) \cdot p(CR)) = \\
 &= (\text{use } c(d_i) > o(CR)) \\
 &\quad g(d_i) \cdot c(d_i) \cdot \mathfrak{P}_i + cr \cdot o(CR) \cdot p(CR).
 \end{aligned}$$

$$\begin{aligned}
 \mathfrak{P}_i &= \theta \circ \partial_H(F_{i+1} \| p(d_i)P) = & (2) \\
 &= \theta(g(d_{i+1}) \cdot \partial_H(s(d_{i+1})F_{i+2} \| p(d_i)P) + \\
 &\quad + cr \cdot \partial_H(p(d_i)P) + \\
 &\quad + p(d_i) \cdot \partial_H(F_{i+1} \| P)) = \\
 &= g(d_{i+1}) \cdot \theta(p(d_i) \cdot \partial_H(s(d_{i+1})F_{i+2} \| P)) + \\
 &\quad + cr \cdot p(d_i) \cdot o(CR) \cdot p(CR) + \\
 &\quad + p(d_i) \cdot \mathfrak{F}_{i+1} = \\
 &= g(d_{i+1}) \cdot p(d_i) \cdot \theta(c(d_{i+1}) \cdot \partial_H(F_{i+2} \| p(d_{i+1})P) + \\
 &\quad + o(CR) \cdot \partial_H(s(d_{i+1})F_{i+2} \| p(CR))) + \\
 &\quad + cr \cdot p(d_i) \cdot o(CR) \cdot p(CR) + p(d_i) \cdot \mathfrak{F}_{i+1} = \\
 &= (\text{use } c(d_{i+1}) > o(CR)) \\
 &\quad g(d_{i+1}) \cdot p(d_i) \cdot c(d_{i+1}) \cdot \mathfrak{P}_{i+1} + \\
 &\quad + cr \cdot p(d_i) \cdot o(CR) \cdot p(CR) + p(d_i) \cdot \mathfrak{F}_{i+1}.
 \end{aligned}$$

State transition diagram:

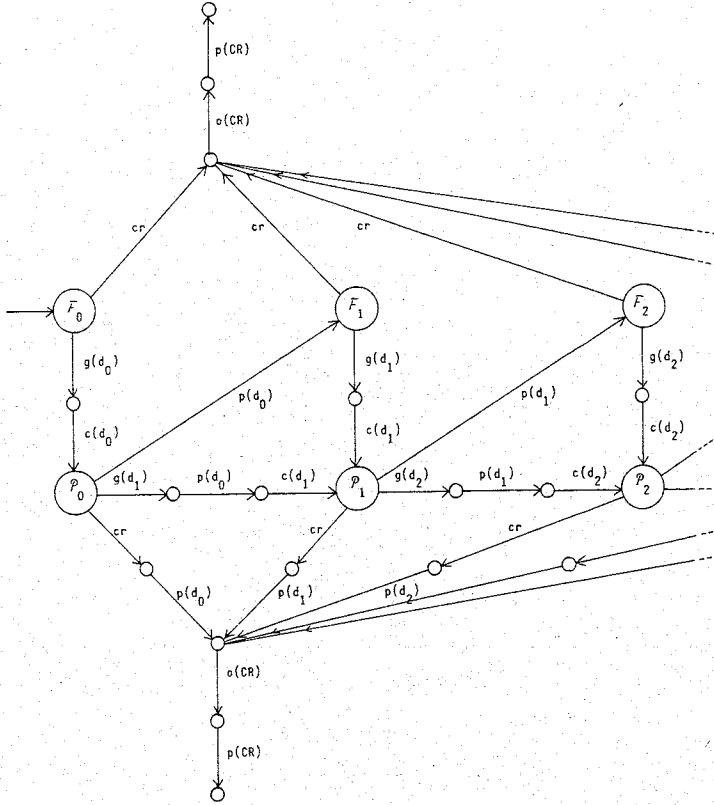


fig.27

3.7 Now we want to focus on the printing actions, in the system just described, and abstract from the other actions. This will give an easy equation for the system. The tool to carry out abstractions is ACP_τ , which is ACP with an abstraction operator τ_I and silent steps τ (see (Bergstra & Klop [2])).

Since we do not want to mix ACP_θ and ACP_τ , we will assume that all θ and \triangleleft are eliminated from terms like \mathfrak{F}_i and \mathfrak{P}_i (possible by 2.3.iii), so that they become ACP-terms. The axiom system ACP_τ is presented on the next page. (table 4)

3.8 Abstraction Define $I = \{c(d), g(d) \mid d \in D\} \cup \{cr, o(CR)\}$ and look at

$$\tau_I \circ \theta \circ \partial_H (F \parallel P)$$

From theorem 3.6, we obtain the following equations:

$$1) \tau_I(\mathfrak{F}_i) = \tau(\tau \tau_I(\mathfrak{P}_i) + \tau p(CR))$$

$$2) \tau_I(\mathfrak{P}_i) = \tau(\tau p(d_i) \tau_I(\mathfrak{P}_{i+1}) + \tau p(d_i) p(CR) + p(d_i) \tau_I(\mathfrak{F}_{i+1})).$$

ACP_τ

$x + y = y + x$	A1	$x\tau = x$	T1
$x + (y + z) = (x + y) + z$	A2	$\tau x + x = \tau x$	T2
$x + x = x$	A3	$a(\tau x + y) = a(\tau x + y) + ax$	T3
$(x + y)z = xz + yz$	A4		
$(xy)z = x(yz)$	A5		
$x + \delta = x$	A6		
$\delta x = \delta$	A7		
$a b = b a$	C1		
$(a b) c = a (b c)$	C2		
$\delta a = \delta$	C3		
$x y = x _y + y _x + x y$	CM1		
$a _x = ax$	CM2	$\tau _x = \tau x$	TM1
$(ax) _y = a(x _y)$	CM3	$(\tau x) _y = \tau(x _y)$	TM2
$(x + y) _z = x _z + y _z$	CM4	$\tau x = \delta$	TC1
$(ax) b = (a b)x$	CM5	$x \tau = \delta$	TC2
$a (bx) = (a b)x$	CM6	$(\tau x) y = x y$	TC3
$(ax) (by) = (a b)(x _y)$	CM7	$x (\tau y) = x y$	TC4
$(x + y) z = x z + y z$	CM8		
$x (y + z) = x y + x z$	CM9		
		$\partial_H(\tau) = \tau$	DT
		$\tau_I(\tau) = \tau$	TI1
$\partial_H(a) = a$ if $a \notin H$	D1	$\tau_I(a) = a$ if $a \notin I$	TI2
$\partial_H(a) = \delta$ if $a \in H$	D2	$\tau_I(a) = \tau$ if $a \in I$	TI3
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3	$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$	TI4
$\partial_H(xy) = \partial_H(x)\partial_H(y)$	D4	$\tau_I(xy) = \tau_I(x)\tau_I(y)$	TI5

3.9 Example 3 Let us now modify the previous example by changing the priority ordering in the following way:

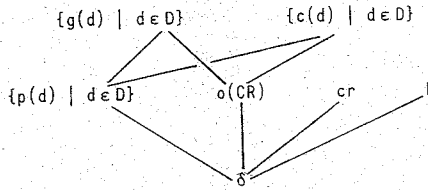


fig.28

This expresses that in real time the 'internal' actions $g(d)$ and $c(d)$ will always precede the external action $p(d)$. This is a simplifying assumption about the real time behaviour of the system and its environment.

Now theorem 3.6 turns into:

3.10 Theorem (abbreviations as in 3.6):

$$1) \mathcal{G}_i = g(d_i) \cdot c(d_i) \mathcal{G}_i + cr \cdot o(CR) \cdot p(CR)$$

$$2) \mathcal{G}_i = g(d_{i+1}) \cdot p(d_i) \cdot c(d_{i+1}) \mathcal{G}_{i+1} + cr \cdot p(d_i) \cdot o(CR) \cdot p(CR). \quad (i \geq 0).$$

Now \mathcal{G}_i has disappeared from the second equation, so this simplifies to:

$$1'): \mathcal{G}_0 = g(d_0) \cdot c(d_0) \mathcal{G}_0 + cr \cdot o(CR) \cdot p(CR)$$

2 as above, and the following state transition diagram (compare with the complicated diagram in 3.6)

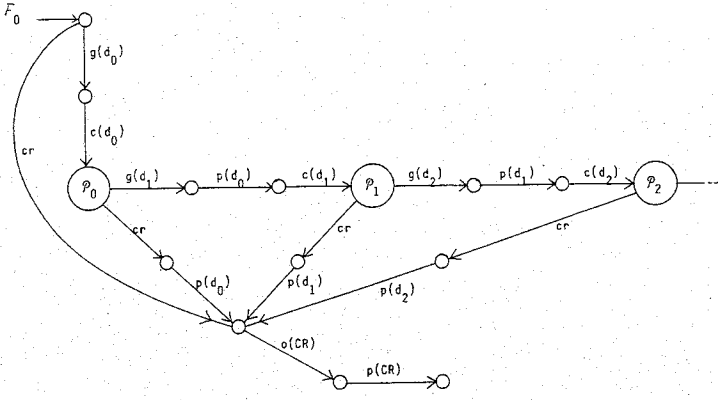


fig.29

When we do abstraction as in 3.8, we get

- 1') $\tau_I(\mathcal{G}_0) = \tau(\tau\tau_I(\mathcal{G}_0) + \tau p(CR))$
- 2) $\tau_I(\mathcal{G}_i) = \tau(\tau p(d_i)\tau_I(\mathcal{G}_{i+1}) + \tau p(d_i)p(CR))$

4. Example: a toy distributed system

4.1 Set-up

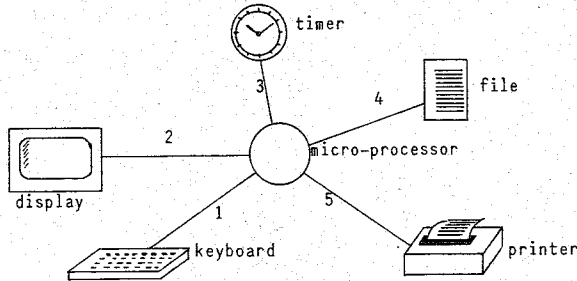


fig.30

Description: At a command from the key-board, a word (string of data) will be released from a file holding infinitely many words, and sent to the printer. Then the word can be printed symbol by symbol. At a signal from the timer, the status of the printer will be requested and the answer displayed. Printing can be interrupted by a BREAK from the key-board.

4.2 Aim We want to give a description of this system, using process algebra. First we will give recursive equations defining each component of the system. Then we will look at the free merge of these processes, will encapsulate unsuccessful communications and give priority to interrupts. Next, we will give recursive equations for the whole system, and then, in order to focus on certain aspects of the system, we will abstract from other elements. We present three ways of doing that.

4.3 Description of components, using state transition diagrams (or process graphs).

4.3.1 Keyboard K. The key-board can generate a message, and send it along channel 1. After that, it is back in its original state.

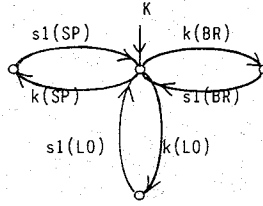


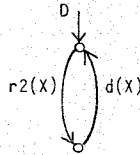
fig.31

actions of K : 1. $k(BR)$ = key in BREAK (will stop printing); 2. $k(LO)$ = key in LOAD (will get a word from the file and load it into the memory of the printer); 3. $k(SP)$ = key in START PRINTING (will cause the printer to start); 4, 5, 6: $s1(BR)$, $s1(LO)$, $s1(SP)$ = send BR , LO , SP along channel 1.

Recursive equation for K :

$$K = (k(BR)s1(BR) + k(LO)s1(LO) + k(SP)s1(SP))K.$$

4.3.2 Display D . The display can receive a message along channel 2 and display it. After that, it is back in its original state.



$$X = PE, PW, PP, PL, PD, PR, FE, ND, NR$$

(the diagram actually consists of 9 loops as shown)

fig.32

actions of D : 1. $d(PE)$ = display: PRINTER ERROR

2. $d(PW)$ = display: PRINTER WAITING

3. $d(PP)$ = display: PRINTER PRINTING

4. $d(PR)$ = display: PRINTER READY.

5. $d(PL)$ = display: PRINTER LOADED, PLEASE KEY IN SP .

6. $d(PD)$ = display: PRINTER DONE, PLEASE KEY IN LOAD.

7. $d(FE)$ = display: FILE ERROR

8. $d(ND)$ = display: PRINTER NOT DONE, PLEASE WAIT

9. $d(NR)$ = display: PRINTER NOT READY, PLEASE WAIT

10-18. $r2(X)$ = receive message X along channel 2 ($X = PE, PW, PP, PR, PL, PD, FE, ND, NR$) Here $PW, PP, PR(2-4)$ are status reports from the printer; $PL, PD(5,6)$ are messages from the printer that it has completed a certain phase; $PE, FE(1,7)$ are error messages, generated when file or printer do not respond; $ND, NR(8,9)$, are messages, generated, when LO or SP is keyed in too early.

Recursive equation for D :

$$D = \left(\sum_{\substack{X = PE, PW, PP, PR, \\ PL, PD, FE, ND, NR}} r2(X)d(X) \right) D.$$

4.3.3 Timer T . The timer can tick, and then it sends a message along channel 3. After that, it is back in its original state.

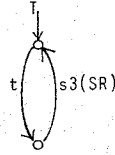


fig.33

actions of T : 1. $t = \text{TICK}$;

2. $s3(SR) = \text{send message STATUS REPORT? along channel 3 (will ask for a status report of the printer)}$.

Recursive equation for T :

$$T = t.s3(SR).T$$

4.3.4 File F . The file holds infinitely many words of length N . Say \mathbb{D} is a finite set of data (maybe containing a blank), then we let \mathbb{D}^N stand for the set of all words of length N . At the request LOAD , F will release the next word and send it to the printer.

If $F = \langle x_1, x_2, \dots \rangle$ (the original state), we put $F_i = \langle x_{i+1}, x_{i+2}, \dots \rangle$ ($i \in \mathbb{N}$, the state after releasing i words (the x_i are in \mathbb{D}^N).

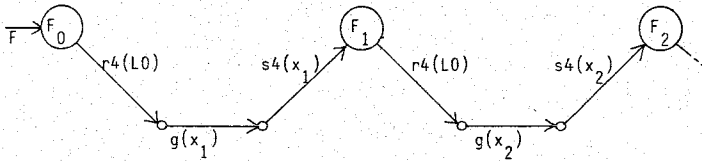


fig.34

In this diagram, we use the following convention: if, inside a node we put a symbol S , then by S we mean the subgraph with S as root.

actions of F :

1. $r4(LO) = \text{receive message LOAD along channel 4}$;

2. $g(x) = \text{get the next word } x \in \mathbb{D}^N \text{ from the file}$

3. $s4(x) = \text{send } x \in \mathbb{D}^N \text{ along channel 4}$.

Equations for F :

$$F = F_0$$

$$F_i = r4(LO).g(x_{i+1}).s4(x_{i+1}).F_{i+1} \quad (i \in \mathbb{N}).$$

4.3.5 Printer P

The printer has three basic states:

1. waiting, with printing queue empty (P_e);

2. ready, with printing queue a word $x \in \mathbb{D}^N$ (printing has not started yet) (R_x);

3. printing, with printing queue a word $x \in \mathbb{D}^{\leq N} = \bigcup_{n=1, \dots, N} \mathbb{D}^n$ (P_x).

The following is a state transition diagram, using a word $x = d_1 \dots d_N \in \mathbb{D}^N$.

vi) $r2(PE) \mid s2(PE) = r2(FE) \mid s2(FE) = em$ (error message)
 vii) for $x \in \mathbb{D}^N$

$$r4(x) \mid s4(x) = r5(x) \mid s5(x) = sd(x) \text{ (send data)}$$

viii) all other communications (i.e. those not defined by i-vii above or rule C1) are δ .

4.5 Then, we want to throw away (encapsulate) all unsuccessful communications, so if we define H to be the set of all $ri(X)$ and all $si(X)$ ($i = 1, 2, 3, 4, 5; X$ a two-letter code or a word in \mathbb{D}^N), we want to look at

$$\partial_H(K \parallel D \parallel T \parallel F \parallel P \parallel M)$$

4.6 Priorities We will define priorities in a certain way, so that strings of actions that belong together, will be executed together, so that the formulation of theorem 4.8 becomes readable. The principles behind this formulation are explained in section 3.

Define $E = \{t, k(BR), k(LO), k(SP)\} \cup \{p(d) \mid d \in D\}$ ('external' actions) and

$$C = \{pm, br, lo, sp, sr\} \cup \{sd(x), g(x) \mid x \in \mathbb{D}^N\}$$

$$\cup \{d(X) \mid X = PE, PW, PP, PR, PL, PD, FE, ND, NR\}$$

(communication actions and other 'internal' actions), then we have the following picture:

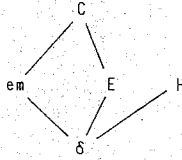


fig.37

(conventions as explained in 3.2).

If θ is defined with respect to this partial ordering, we look at

$$\mathfrak{W}_0 = \theta \circ \partial_H(K \parallel D \parallel T \parallel F \parallel P \parallel M)$$

4.7 Now we will prove some recursive equations that hold for the whole system \mathfrak{W}_0 . First some abbreviations:

$$\mathfrak{W}_i = \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_e \parallel M_w) \quad (i \in \mathbb{N})$$

(printer is waiting and i words have been handled)

$$\mathfrak{R}_i = \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel R_x \parallel M_r) \quad (i \geq 1)$$

(printer is loaded, the i -th word is being handled)

$$\mathfrak{P}_i^n = \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_x \parallel M_p) \quad (i \geq 1, n = 1, \dots, N,$$

x is (the tail of) x_i of length n).

(printer is printing the i -th word, still n characters to go)

$$\mathfrak{P}_i^0 = \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel (s5(PD)P) \parallel M_p) \quad (i \geq 1)$$

(printer has just printed the last character of the i -th word)

4.8 Theorem:

1. $\mathfrak{W}_i = k(BR) \cdot br \cdot \mathfrak{W}_i +$
 $+ k(LO) \cdot lo \cdot g(x_{i+1}) \cdot sd(x_{i+1}) \cdot pm \cdot pm \cdot d(PL) \cdot \mathfrak{R}_{i+1} +$
 $+ k(SP) \cdot sp \cdot pm \cdot d(NR) \cdot \mathfrak{W}_i +$
 $+ t \cdot sr \cdot sr \cdot sr \cdot d(PW) \cdot \mathfrak{W}_i. \quad (i \geq 0)$

2. $\mathfrak{R}_i = k(BR) \cdot br \cdot br \cdot \mathfrak{U}_i +$
 $+ k(LO) \cdot lo \cdot pm \cdot d(ND) \cdot \mathfrak{R}_i +$
 $+ k(SP) \cdot sp \cdot sp \cdot \mathfrak{P}_i^N +$
 $+ t \cdot sr \cdot sr \cdot sr \cdot sr \cdot d(PR) \cdot \mathfrak{R}_i. (i \geq 1)$
3. $\mathfrak{P}_i^n = k(BR) \cdot br \cdot br \cdot \mathfrak{U}_i +$
 $+ k(LO) \cdot lo \cdot pm \cdot d(ND) \cdot \mathfrak{P}_i^n +$
 $+ k(SP) \cdot sp \cdot sp \cdot \mathfrak{P}_i^n +$
 $+ t \cdot sr \cdot sr \cdot sr \cdot sr \cdot d(PP) \cdot \mathfrak{P}_i^n +$
 $+ p(d) \mathfrak{P}_i^{n-1} (i \geq 1, n = 1, \dots, N, d \text{ is the } (N+1-n)\text{th character of } x_i)$
4. $\mathfrak{P}_i^0 = pm \cdot pm \cdot d(PD) \cdot \mathfrak{U}_i.$

Proof: We use ET as in the proof of 3.3.

1. Let $i \geq 0$

$$\begin{aligned}
 \mathfrak{U}_i &= \theta \circ \partial_H(K \| D \| T \| F_i \| P_e \| M_w) = \\
 &\quad \theta \circ \partial_H(k(BR) \cdot ((s \ 1(BR)K) \| D \| T \| F_i \| P_e \| M_w) + \\
 &\quad + k(LO) \cdot ((s \ 1(LO)K) \| D \| T \| F_i \| P_e \| M_w) + \\
 &\quad + k(SP) \cdot ((s \ 1(SP)K) \| D \| T \| F_i \| P_e \| M_w) + \\
 &\quad + r \ 2(PE) \cdot (K \| (d(PE)D) \| T \| F_i \| P_e \| M_w) + \\
 &\quad + r \ 2(PW) \cdot (K \| (d(PW)D) \| T \| F_i \| P_e \| M_w) + \\
 &\quad + r \ 2(PP) \cdot (K \| (d(PP)D) \| T \| F_i \| P_e \| M_w) + \\
 &\quad \quad \quad \text{--- (six more } r \ 2(X) \text{) ---} \\
 &\quad + r \ 2(NR) \cdot (K \| (d(NR)D) \| T \| F_i \| P_e \| M_w) + \\
 &\quad + t \cdot (K \| D \| (s \ 3(SR)T) \| F_i \| P_e \| M_w) + \\
 &\quad + r \ 4(LO) \cdot (K \| D \| T \| (g(x_{i+1}) s \ 4(x_{i+1}) F_{i+1}) \| P_e \| M_w) + \\
 &\quad + \sum_{x \in \mathbf{D}^N} r \ 5(x) \cdot (K \| D \| T \| F_i \| (s \ 5(PL)R_x) \| M_w) + \\
 &\quad + r \ 5(SR) \cdot (K \| D \| T \| F_i \| (s \ 5(PW)P_d) \| M_w) + \\
 &\quad + r \ 5(BR) \cdot (K \| D \| T \| F_i \| P_e \| M_w) + \\
 &\quad + r \ 3(SR) \cdot (K \| D \| T \| F_i \| P_e \| (s \ 5(SR)(s \ 2(PE) + r \ 5(PW)s \ 2(PW))M_w) + \\
 &\quad + r \ 1(SP) \cdot (K \| D \| T \| F_i \| P_e \| (s \ 2(NR)M_w)) + \\
 &\quad + r \ 1(BR) \cdot (K \| D \| T \| F_i \| P_e \| (s \ 5(BR)M_w)) + \\
 &\quad + r \ 1(LO) \cdot (K \| D \| T \| F_i \| P_e \| (s \ 4(LO)(s \ 2(FE)M_w + \dots M_r))) + \\
 &\quad + \delta) \text{ (no communications possible)} = \\
 &= \theta(k(BR) \cdot \partial_H((s \ 1(BR)K) \| D \| T \| F_i \| P_e \| M_w) + \\
 &\quad + k(LO) \cdot \partial_H((s \ 1(LO)K) \| D \| T \| F_i \| P_e \| M_w) + \\
 &\quad + k(SP) \cdot \partial_H((s \ 1(SP)K) \| D \| T \| F_i \| P_e \| M_w) + \\
 &\quad + t \cdot \partial_H(K \| D \| (s \ 3(SR)T) \| F_i \| P_e \| M_w) = \\
 &= k(BR) \cdot \theta(t \cdot \partial_H((s \ 1(BR)K) \| D \| (s \ 3(SR)T) \| F_i \| P_e \| M_w) + \\
 &\quad + (s \ 1(BR) \mid r \ 1(BR)) \cdot \partial_H(K \| D \| T \| F_i \| P_e \| (s \ 5(BR)M_w))
 \end{aligned}$$

$$\begin{aligned}
& + k(LO) \cdot \theta(t \cdot \partial_H((s \ 1(LO)K) \| D \| (s \ 3(SR)T) \| F_i \| P_e \| M_w) + \\
& + (s \ 1(LO) | r \ 1(LO)) \cdot \partial_H(K \| D \| T \| F_i \| P_e \| (s \ 4(LO)(\dots M_w + \dots M_r))) + \\
& + k(SP) \cdot \theta(t \cdot \partial_H((s \ 1(SP)K) \| D \| (s \ 3(SR)T) \| F_i \| P_e \| M_w) + \\
& + (s \ 1(SP) | r \ 1(SP)) \cdot \partial_H(K \| D \| T \| F_i \| P_e \| (s \ 2(NR)M_w)) + \\
& + t \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + \\
& + (s \ 3(SR) | r \ 3(SR)) \cdot \partial_H(K \| D \| T \| F_i \| P_e \| (s \ 5(SR)(\dots)M_w)) = \\
& = \text{(by 4.6)} \\
& k(BR) \cdot br \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
& + (s \ 5(BR) | r \ 5(BR)) \cdot \partial_H(K \| D \| T \| F_i \| P_e \| M_w) + \\
& + k(LO) \cdot lo \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
& + (s \ 4(LO) | r \ 4(LO)) \cdot \partial_H(K \| D \| T \| (g(x_{i+1})s \ 4(x_{i+1})F_{i+1}) \| P_e \| \\
& \quad \| (s \ 2(FE)M_w + \sum_x r \ 4(x) \dots)) + \\
& + k(SP) \cdot sp \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
& + (s \ 2(NR) | r \ 2(NR)) \cdot \partial_H(K \| (d(NR)D) \| T \| F_i \| P_e \| M_w) + \\
& + t \cdot sr \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
& + (s \ 5(SR) | r \ 5(SR)) \cdot \partial_H(K \| D \| T \| F_i \| (s \ 5(PW)P_e) \| (s \ 2(PE) + \\
& \quad + r \ 5(PW)s \ 2(PW))M_w) = \\
& = k(BR) \cdot br \cdot br \cdot \mathfrak{U}_i^{(1)} + \\
& + k(LO) \cdot lo \cdot lo \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
& + g(x_{i+1}) \cdot \partial_H(K \| D \| T \| s \ 4(x_{i+1})F_{i+1} \| P_e \| (s \ 2(FE) \dots + \sum_x r \ 4(x) \dots)) + \\
& + (s \ 2(FE) | r \ 2(FE)) \cdot \partial_H(\dots)^{(2)} + \\
& + k(SP) \cdot sp \cdot pm \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
& + d(NR) \cdot \partial_H(K \| D \| T \| F_i \| P_e \| M_w))^{(3)} + \\
& + t \cdot sr \cdot sr \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
& + (s \ 5(PW) | r \ 5(PW)) \cdot \partial_H(K \| D \| T \| F_i \| P_e \| s \ 2(PW)M_w) + \\
& + (s \ 2(PE) | r \ 2(PE)) \cdot \partial_H(\dots)^{(4)}.
\end{aligned}$$

Now term 1 is right, and it is easy to see that term 3 is also right. Therefore, we only continue with terms 2 and 4.

$$\begin{aligned}
2 & = k(LO) \cdot lo \cdot lo \cdot g(x_{i+1}) \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
& + (s \ 4(x_{i+1}) | r \ 4(x_{i+1})) \cdot \partial_H(K \| D \| T \| F_{i+1} \| P_e \| s \ 5(x_{i+1})(\dots)) + \\
& + (s \ 2(FE) | r \ 2(FE)) \cdot \partial_H(\dots) = \\
& = k(LO) \cdot lo \cdot lo \cdot g(x_{i+1}) \cdot sd(x_{i+1}) \cdot \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
& + (s \ 5(x_{i+1}) | r \ 5(x_{i+1})) \cdot \partial_H(K \| D \| T \| F_{i+1} \| s \ 5(PL)R_{x_{i+1}} \| \\
& \quad \| (s \ 2(PE)M_w + r \ 5(PL)s \ 2(PL)M_r)) = \\
& = k(LO) \cdot lo \cdot lo \cdot g(x_{i+1}) \cdot sd(x_{i+1}) \cdot sd(x_{i+1}) \cdot \theta(\dots + \dots + \dots + \dots + \\
& + (s \ 5(PL) | r \ 5(PL)) \cdot \partial_H(K \| D \| T \| F_{i+1} \| R_{x_{i+1}} \| s \ 2(PL)M_r) + \\
& + (s \ 2(PE) | r \ 2(PE)) \cdot \partial_H(\dots) =
\end{aligned}$$

$$\begin{aligned}
&= k(LO) \succ lo \cdot lo \cdot g(x_{i+1}) \succ sd(x_{i+1}) \succ pm \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ (s2(PL) \mid r2(PL)) \succ \partial_H(K \parallel d(PL)D \parallel T \parallel F_{i+1} \parallel R_{x_{i+1}} \parallel M_r) = \\
&= k(LO) \succ lo \cdot lo \cdot g(x_{i+1}) \succ sd(x_{i+1}) \succ sd(x_{i+1}) \succ pm \cdot pm \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ d(PL) \succ \partial_H(K \parallel D \parallel T \parallel F_{i+1} \parallel R_{x_{i+1}} \parallel M_r) = \\
&= k(LO) \succ lo \cdot lo \cdot g(x_{i+1}) \succ sd(x_{i+1}) \succ sd(x_{i+1}) \succ pm \cdot pm \cdot d(PL) \succ \mathfrak{R}_{i+1};
\end{aligned}$$

$$\begin{aligned}
4 &= t \cdot sr \cdot sr \cdot sr \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ (s2(PW) \mid r2(PW)) \succ \partial_H(K \parallel d(PW)D \parallel T \parallel F_i \parallel P_\epsilon \parallel M_w)) = \\
&= t \cdot sr \cdot sr \cdot sr \cdot sr \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ d(PW) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_\epsilon \parallel M_w)) = \\
&= t \cdot sr \cdot sr \cdot sr \cdot sr \cdot d(PW) \succ \mathfrak{R}_i
\end{aligned}$$

This finishes the proof of 1.

2. Let $i \geq 1$.

$$\begin{aligned}
\mathfrak{R}_i &= \theta \cdot \partial_H(K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel M_r) = \\
&\theta(k(BR) \succ \partial_H(s1(BR)K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel M_r) + \\
&+ k(LO) \succ \partial_H(s1(LO)K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel M_r) + \\
&+ k(SP) \succ \partial_H(s1(SP)K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel M_r) + \\
&+ t \cdot \partial_H(K \parallel D \parallel s3(SR)T \parallel F_i \parallel R_{x_i} \parallel M_r)) = \\
&= k(BR) \succ \theta(t \cdot \dots + (s1(BR) \mid r1(BR)) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel s5(BR)M_w)) + \\
&+ k(LO) \succ \theta(t \cdot \dots + (s1(LO) \mid r1(LO)) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel s2(ND)M_r)) + \\
&+ k(SP) \succ \theta(t \cdot \dots + (s1(SP) \mid r1(SP)) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel s5(SP)M_p)) + \\
&+ t \cdot \theta(\dots + \dots + \dots + (s3(SR) \mid r3(SR)) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel s5(SR)(\dots)M_r) = \\
&= k(BR) \succ br \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ (s5(BR) \mid r5(BR)) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_\epsilon \parallel M_w)) + \\
&+ k(LO) \succ lo \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ (s2(ND) \mid r2(ND)) \succ \partial_H(K \parallel d(ND)D \parallel T \parallel F_i \parallel R_{x_i} \parallel M_r)) + \\
&+ k(SP) \succ sp \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ (s5(SP) \mid r5(SP)) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_{x_i} \parallel M_p)) + \\
&+ t \cdot sr \cdot \theta(\dots + \dots + \dots + \dots + (s5(SR) \mid r5(SR)) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel \\
&\quad s5(PR)R_{x_i} \parallel (s2(PE) + r5(PR)s2(PR))M_r)) = \\
&= k(BR) \succ br \cdot br \cdot \mathfrak{R}_i + \\
&+ k(LO) \succ lo \cdot pm \cdot \theta(\dots + \dots + \dots + \dots + d(ND) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel M_r)) + \\
&+ k(SP) \succ sp \cdot sp \cdot \mathfrak{R}_i^N + \\
&+ t \cdot sr \cdot sr \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ (s5(PR) \mid r5(PR)) \succ \partial_H(K \parallel D \parallel T \parallel F_i \parallel R_{x_i} \parallel s2(PR)M_r) + \\
&+ (s2(PE) \mid r2(PE)) \succ \partial_H(\dots)) =
\end{aligned}$$

$$\begin{aligned}
&= k(BR) \cdot br \cdot br \cdot \mathfrak{U}_i + \\
&+ k(LO) \cdot lo \cdot pm \cdot d(ND) \cdot \mathfrak{R}_i + \\
&+ k(SP) \cdot sp \cdot sp \cdot \mathfrak{P}_i^N + \\
&+ t \cdot sr \cdot sr \cdot sr \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ (s2(PR) \mid r2(PR)) \cdot \partial_H(K \parallel d(PR)D \parallel T \parallel F_i \parallel R_x \parallel M_r)),
\end{aligned}$$

and we can finish as in 1.

3. Let $i \geq 1$, $n = 1, \dots, N$ and $x = d_1 \dots d_n$ is (the tail of) x_i . Then d_1 is the $(N+1-n)^{th}$ character of x_i .

case 1: $n > 1$. Then

$$\begin{aligned}
\mathfrak{P}_i^n &= \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_x \parallel M_p) = \\
&= \theta(k(BR) \cdot \partial_H(s1(BR)K \parallel D \parallel T \parallel F_i \parallel P_x \parallel M_p) + \\
&+ k(LO) \cdot \partial_H(s1(LO)K \parallel D \parallel T \parallel F_i \parallel P_x \parallel M_p) + \\
&+ k(SP) \cdot \partial_H(s1(SP)K \parallel D \parallel T \parallel F_i \parallel P_x \parallel M_p) + \\
&+ t \cdot \partial_H(K \parallel D \parallel s3(SR)T \parallel F_i \parallel P_x \parallel M_p) + \\
&+ p(d_1) \cdot \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_{d_1 \dots d_n} \parallel M_p)) \\
&= k(BR)(s1(BR) \mid r1(BR)) \cdot \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_x \parallel s5(BR)M_w) + \\
&+ k(LO)(s1(LO) \mid r1(LO)) \cdot \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_x \parallel s2(ND)M_p) + \\
&+ k(SP)(s1(SP) \mid r1(SP)) \cdot \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_x \parallel M_p) + \\
&+ t(s3(SR) \mid r3(SR)) \cdot \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_x \parallel s5(SR) \dots M_p) \\
&+ p(d_1) \mathfrak{P}_i^{n-1}.
\end{aligned}$$

Again it is easy to finish the proof.

case 2: $n = 1$. The first four terms are the same, so

$$\begin{aligned}
\mathfrak{P}_i^1 &= \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel P_{d_1} \parallel M_p) = \\
&= k(BR) \cdot br \cdot br \cdot \mathfrak{U}_i + k(LO) \cdot lo \cdot pm \cdot d(ND) \cdot \mathfrak{P}_i^1 + \\
&+ k(SP) \cdot sp \cdot \mathfrak{P}_i^1 + t \cdot sr \cdot sr \cdot sr \cdot d(PP) \cdot \mathfrak{P}_i^1 + \\
&+ p(d_1) \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel s5(PD)P_e \parallel M_p) \quad (\text{by 4.3.5.iv}) = \\
&= \dots + \dots + \dots + \dots + \\
&+ p(d_1) \mathfrak{P}_i^0
\end{aligned}$$

4. Let $i \geq 1$.

$$\begin{aligned}
\mathfrak{P}_i^0 &= \theta \circ \partial_H(K \parallel D \parallel T \parallel F_i \parallel s5(PD)P \parallel M_p) = \\
&= \theta(k(BR) \dots + k(LO) \dots + k(SP) \dots + t \dots + \\
&+ (s5(PD) \mid r5(PD)) \cdot \partial_H(K \parallel D \parallel T \parallel F_i \parallel P \parallel s2(PD)M_w)) = \\
&= pm \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ (s2(PD) \mid r2(PD)) \cdot \partial_H(K \parallel d(PD)D \parallel T \parallel F_i \parallel P \parallel M_w)) = \\
&= pm \cdot pm \cdot \theta(\dots + \dots + \dots + \dots + \\
&+ d(PD) \cdot \partial_H(K \parallel D \parallel T \parallel F_i \parallel P \parallel M)) = \\
&= pm \cdot pm \cdot d(PD) \cdot \mathfrak{U}_i.
\end{aligned}$$

This completes the proof of theorem 4.8.

4.9 Now we want to focus on certain aspects of the system just described, and abstract from others. We will present three ways of doing this, namely:

1. abstract from all internal steps, focus on key-board, display and printer;
2. abstract from all internal steps, focus on file and printer (i/o view);
3. get easier equations for 2 by hiding interrupts.

As in 3.8, we work in ACP_{τ} , but now we need an extra abstraction rule, since infinite τ -paths can occur. The rule is explained in 4.10.

4.10 Koomen's fair abstraction rule (KFAR) (see Bergstra & Klop [3]).

This rule allows us to compute $\tau_I(X)$ for certain X , thereby expressing the fact that certain steps in I will be fairly scheduled in such a way that eventually a step outside I is performed. Formally,

$$(KFAR) \frac{\forall n \in \mathbb{Z}_k \quad X_n = i_n \cdot X_{n+1} + Y_n \quad (i_n \in I)}{\tau_I(X_n) = \tau \cdot \tau_I(Y_0 + \dots + Y_{k-1})}$$

Here $\mathbb{Z}_k = \{0, \dots, k-1\}$ and addition in subscripts works modulo k . For the use of KFAR, also see [3].

4.11 Define $I = \{t, br, sr, lo, sp, pm, em\} \cup$

$$\cup \{g(x) \mid x \in \mathbb{D}^N\} \cup \{sd(x) \mid x \in \mathbb{D}^N\} \cup \\ \cup \{d(PW), d(PR), d(PP)\}.$$

This means we abstract from the timer, the file, all communications and the status reports. Now applying τ_I to the equations of theorem 4.8, and using KFAR, gives:

1. $\tau_I(\mathbb{W}_i) = \tau(k(BR) + k(SP)d(NR))\tau_I(\mathbb{W}_i) +$
 $+ k(LO)d(PL)\tau_I(\mathbb{W}_{i+1})) \quad (i \geq 0)$
2. $\tau_I(\mathbb{R}_i) = \tau(k(BR)\tau_I(\mathbb{W}_i) +$
 $+ k(LO)d(ND)\tau_I(\mathbb{R}_i) +$
 $+ k(SP)\tau_I(\mathbb{P}_i^N)) \quad (i \geq 1)$
3. $\tau_I(\mathbb{P}_i^n) = \tau(k(BR)\tau_I(\mathbb{W}_i) +$
 $+ k(LO)d(ND)\tau_I(\mathbb{P}_i^n) +$
 $+ k(SP)\tau_I(\mathbb{P}_i^n) +$
 $+ p(d)\mathbb{P}_i^{n-1}) \quad (i \geq 0, n = 1, \dots, N,$
 $d \text{ is the } (N+1-n) \text{th character of } x_i)$
4. $\tau_I(\mathbb{P}_i^0) = \tau d(PD)\tau_I(\mathbb{W}_i).$

We can make the following state transition diagram of $\tau_I(\mathbb{W}_0)$. (Suppose $x_1 = d_1 \dots d_N$)

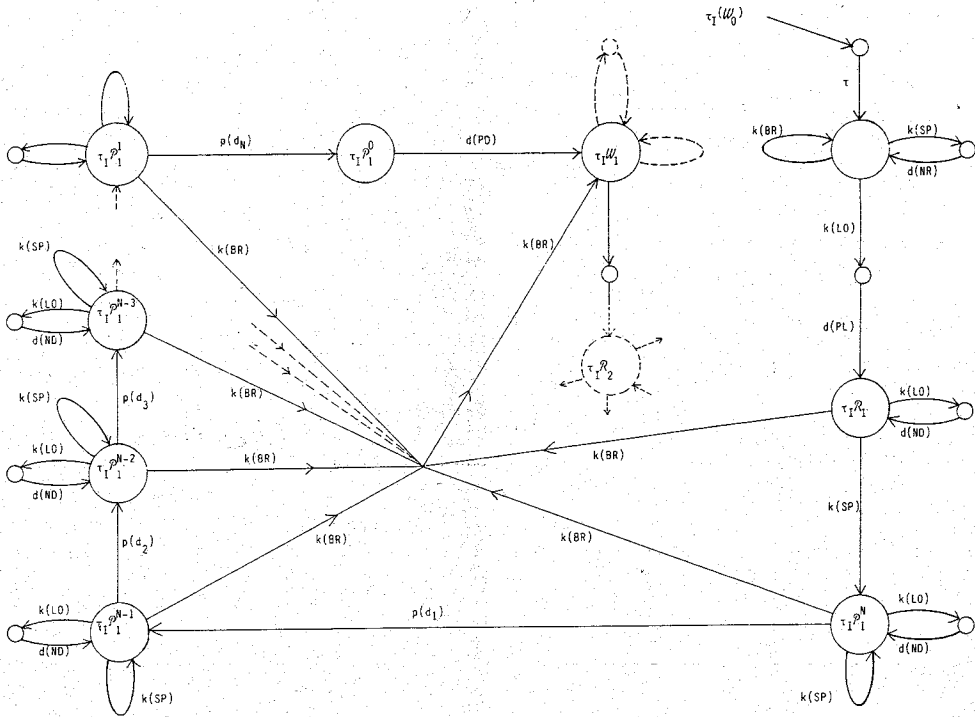


fig.38

4.12 Second version:

Diagram:

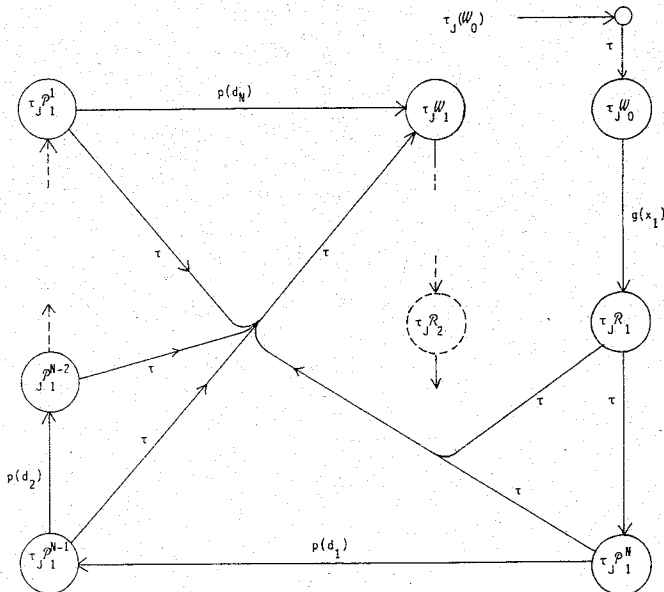


fig.39

$$\begin{aligned}
\text{Define } J &= \{t, br, sr, lo, sp, pm, em\} \cup \\
&\cup \{sd(x) \mid x \in \mathbb{D}^N\} \cup \\
&\cup \{d(PW), d(PR), d(PP), d(PE), d(PL), d(PD), \\
&\quad d(FE), d(ND), d(NR)\} \cup \\
&\cup \{k(LO), k(SP), k(BR)\}.
\end{aligned}$$

This means we abstract from the timer, the display, the keyboard and all communications. Now apply τ_J and use KFAR. We get:

1. $\tau_J(\mathcal{W}_i) = \tau g(x_{i+1}) \tau_J(\mathcal{R}_{i+1}). (i \geq 0).$
2. $\tau_J(\mathcal{R}_i) = \tau(\tau \tau_J(\mathcal{W}_i) + \tau \tau_J(\mathcal{P}_i^N)). (i \geq 1).$
3. $\tau_J(\mathcal{P}_i^n) = \tau(\tau \tau_J(\mathcal{W}_i) + p(d) \tau_J(\mathcal{P}_i^{n-1})). (i \geq 0, n = 1, \dots, N, d \text{ the } (N+1-n)\text{th in } x_i)$
4. $\tau_J(\mathcal{P}_i^0) = \tau \tau_J(\mathcal{W}_i) (i \geq 1).$

4.13 In order to see better, that the input is really that what is printed, we will use the following trick. Define $B = \{k(BR)\}$ and look at

$$\boxed{\tau_J \circ \partial_B(\mathcal{W}_i)}$$

This means that we throw away all paths that have a break, and then do the abstraction of 3.12

Claim: Suppose $x_{i+1} = d_1^i \dots d_N^i (i \geq 0).$

$$\tau_J \circ \partial_B(\mathcal{W}_i) = \tau g(x_{i+1}) p(d_1^i) p(d_2^i) \dots p(d_N^i) \tau_J \circ \partial_B(\mathcal{W}_{i+1}).$$

Proof: we start with the equations of theorem 3.8.

$$\begin{aligned}
\partial_B(\mathcal{W}_i) &= \delta + a_1 a_2 a_3 g(x_{i+1}) a_3 a_4 a_5 \partial_B(\mathcal{R}_{i+1}) + \\
&\quad + a_6 a_7 a_8 \partial_B(\mathcal{W}_i) + a_9 a_{10} a_{10} a_{10} a_{11} \partial_B(\mathcal{W}_i)
\end{aligned}$$

(here the $a_1 - a_{11}$ symbolize internal steps), so by KFAR we have

1. $\tau_J \circ \partial_B(\mathcal{W}_i) = \tau g(x_{i+1}) \tau_J \circ \partial_B(\mathcal{R}_{i+1}).$
Likewise
2. $\tau_J \circ \partial_B(\mathcal{R}_{i+1}) = \tau \tau_J \circ \partial_B(\mathcal{P}_{i+1}^N)$
and
3. $\tau_J \circ \partial_B(\mathcal{P}_{i+1}^n) = \tau p(d_{N+1-n}^i) \tau_J \circ \partial_B(\mathcal{P}_{i+1}^{n-1}) (n = 1, \dots, N)$
and
4. $\tau_J \circ \partial_B(\mathcal{P}_{i+1}^0) = \tau \tau_J \circ \partial_B(\mathcal{W}_{i+1}).$

Combining these gives

$$\begin{aligned}
\tau_J \circ \partial_B(\mathcal{W}_i) &= \tau g(x_{i+1}) \tau_J \circ \partial_B(\mathcal{R}_{i+1}) = \\
&= \tau g(x_{i+1}) \tau \tau_J \circ \partial_B(\mathcal{P}_{i+1}^N) = \\
&= \tau g(x_{i+1}) \tau p(d_1^i) \tau_J \circ \partial_B(\mathcal{P}_{i+1}^{N-1}) = \\
&= \tau g(x_{i+1}) p(d_1^i) \tau p(d_2^i) \tau_J \circ \partial_B(\mathcal{P}_{i+1}^{N-2}) = \\
&= \dots = \\
&= \tau g(x_{i+1}) p(d_1^i) p(d_2^i) \dots \tau p(d_N^i) \tau_J \circ \partial_B(\mathcal{P}_{i+1}^0) = \\
&= \tau g(x_{i+1}) p(d_1^i) p(d_2^i) \dots p(d_N^i) \tau \tau_J \circ \partial_B(\mathcal{W}_{i+1}) = \\
&= \tau g(x_{i+1}) p(d_1^i) p(d_2^i) \dots p(d_N^i) \tau_J \circ \partial_B(\mathcal{W}_{i+1}).
\end{aligned}$$

References

- [1] DE BAKKER, J.W. & J.I. ZUCKER, *Processes and the denotational semantics of concurrency*, Information and Control, 54 (1/2), 1982, pp.70-120.
- [2] BERGSTRÄ, J.A. & J.W. KLOP, *Algebra of communicating processes with abstraction*, Theoretical Comp. Sci. 37 (1), 1985, pp. 77-121.
- [3] BERGSTRÄ, J.A. & J.W. KLOP, *Verification of an alternating bit protocol by means of process algebra*, Report CS-R8404, Centrum voor Wiskunde en Informatica, Amsterdam 1984.
- [4] BERGSTRÄ, J.A. & J.W. KLOP, *Process algebra for synchronous communication*, Information and Control 60 (1/3), 1984, pp. 109-137.
- [5] BERGSTRÄ, J.A. & J.V. TUCKER, *Top-down design and the algebra of communicating processes*, Science of Comp. Programming 5 (2), 1985, pp. 171-199.
- [6] DERSHOWITZ, N., *Orderings for term-rewriting systems*, Theoretical Computer Science 17, 1982, pp. 279-301.
- [7] HUET, G., *Confluent reductions: abstract properties and applications to term rewriting systems*, J. ACM 27 (4), 1980, pp. 797-821.
- [8] KLOP, J.W., *Combinatory reduction systems*, Mathematical Centre Tract 127, Mathematisch Centrum, Amsterdam 1980.
- [9] MILNER, R., *A calculus for communicating systems*, Springer LNCS 92, 1980.
- [10] PARK, D.M.R., *Concurrency and automata on infinite sequences*, Springer LNCS 104, 1981.