stichting
mathematisch
centrum

MC

AFDELING INFORMATICA IW 232/83 JULI
(DEPARTMENT OF COMPUTER SCIENCE)

J.A. BERGSTRA & J.W. KLOP

AN ALGEBRAIC SPECIFICATION METHOD FOR PROCESSES OVER A FINITE ACTION SET

Preprint

kruislaan 413 1098 SJ amsterdam

BIBLIOTHEEK MATHEMATISCH CENTRUM
AMSTERDAM

Printed at the Mathematical Centre, Kruislaan 413, Amsterdam, The Netherlands.

The Mathematical Centre, founded 11 February 1946, is a non-profit institution for the promotion of pure and applied mathematics and computer science. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

1980 Mathematics subject classification: 68B10, 68C01, 68D25, 68F20

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

Copyright © 1983, Mathematisch Centrum, Amsterdam

An algebraic specification method for processes over a finite action set *)

by

J.A. Bergstra & J.W. Klop

ABSTRACT

We combine the techniques of abstract data type specification and of process algebra thus obtaining a flexible technique for process specification, provided a finite action set is used.

KEY WORDS & PHRASES: *concurrency, nondeterministic process, merge, process algebra, state transition system, algebraic specification, computable process*

---

*) This report is not for review as it will be published elsewhere.

INTRODUCTION

The question how to properly embed the concept of a process in a mathematical theory is still open. Several approaches are already well established in the litterature; to mention some:

(i) CCS: MILNER [12]

(ii) CSP: HOARE [11]

(iii) processes and scenarios: BROCK & ACKERMAN [10], PRATT [14]

(iv) trace theories: ARNOLD [1], BACK & MANNILA [2], REM [15].

Recently DE BAKKER & ZUCKER in [3,4] have proposed a denotational semantics of concurrency, based on an underlying mathematical model of concurrency which views infinite processes as Cauchy sequences of processes of finite depth. (A process is of depth n if its executions entail sequences of atomic acts of at most length n.)

This model is quite attractive because of its clear and rich structure. Restricting interest to processes over a finite alphabet A of atomic actions, an algebraic description of the structure of DE BAKKER and ZUCKER was given in BERGSTRA & KLOP [5,6].

In our algebraic format infinite processes occur as projective limits of finite processes. In more detail:

For each $n \in \{1,2,...\}$ there is a structure A_n containing processes of depth n or less. A projection operator $(\ )_{n+1}$ maps A_{n+1} into A_n, essentially $(p)_{n+1}$ just skips in p all actions at depth n+1.

A process p is then a sequence $p = (p_1, p_2, p_3, ...)$ of elements of $(A_1, A_2, A_3, ...)$ with the property that for each n, $(p_{n+1})_n = p_n$. Thus p is identified with a sequence of approximations where always the next approximation refines the previous approximation by introducing structure at one more level.

As operations on processes three operators are indispensable and others can be added:

$+$: *alternative composition (choice)*

$\cdot$: *sequential composition*

$\|$: *parallel composition (merge)*

We use an auxiliary operator $⌊\!⌊$: *left merge*.

The process $p \mathbin{\|\!\_} q$ is like $p \| q$ but insists on taking its first step from p.

The construction of processes, to be explained in more detail in Section 1, proceeds in three stages:

(i) Find a model A_ω of all processes of finite depth over A.

(ii) Construct the family of structures $A_n = A_\omega$ modulo n for $n \geqslant 1$ by taking suitable (homomorphic) projections of A_ω.

(iii) Construct the projective limit of the A_n: $A^\infty = \varprojlim A_n$.

Under the assumption that A^∞ is a suitable domain for process theory we then are faced with a major difficulty:

How to specify a process?

As methods two mechanisms present themselves naturally:

(1) *Explicit description of the projective sequence* $p = (p_1, p_2, \ldots)$.

Note that each A_n is finite and each p_n is a finite combinatorial object which can be written as a term built over $A, +, \cdot, \|, \mathbin{\|\!\_}$.

Example. Let $A = \{a,b\}$, and let

$$p_n = \sum_{\alpha \in A_n} \alpha \quad .$$

Indeed $(p_{n+1})_n = p_n$ for each n. Let $p = \lim p_n$. Clearly p possesses a computable projective sequence and has been properly specified in quite an explicit way.

(2) *Recursive definitions.*

Let $X_i = T_i(X_1, \ldots, X_n)$, $i = 1, \ldots, n$, be a system of guarded equations over $A, +, \cdot, \|, \mathbin{\|\!\_}$.

Such a system has a unique solution in A^∞. Processes that occur in solution vectors are called *recursively definable* in BERGSTRA & KLOP [7], where it was also shown that

(i) recursively defined processes suffice to yield a semantics for CSP processes,

(ii) systems of equations are a stronger definitional mechanism than single equations and

(iii) adding communication to the algebra may strictly extend the class of recursively definable processes.

Both methods (1) and (2) have drawbacks. In (1) the difficulty may be to find an appropriate description of the $(p)_n$. For a counter already this is unpleasant.

Similarly in (2) one may need unreasonably complex systems of equations to specify essentially simple processes.

Therefore we will add a third facility in the present paper. This mechanism views a process as the behaviour of a state transition system. This state transition system is modeled as an algebra and algebraic specification techniques are applied.

(3) *Algebraic specification of state transition systems, followed by behavioural abstraction.*

This method allows a quick specification of processes that can be viewed as appropriate behaviours of transition systems.

The structure of the paper is as follows:

1. PROCESS ALGEBRA, A^∞
2. ALGEBRAIC SPECIFICATION OF STATE TRANSITION SYSTEMS
3. BEHAVIOURAL ABSTRACTION
4. EXAMPLES OF PROCESS SPECIFICATION
5. COMPARISON OF THE RELATIVE POWER OF THE THREE SPECIFICATION MECHANISMS

1. PROCESS ALGEBRA, A^∞

Let A be a finite set of atomic actions. These are taken as constants in a signature Σ_A together with operations $+, \cdot, \|, \lfloor\!\lfloor$. The axiom system PA is then as follows:

$x + y = y + x$	A1
$x + (y + z) = (x + y) + z$	A2
$x + x = x$	A3
$(x + y).z = x.z + y.z$	A4
$(x.y).z = x.(y.z)$	A5
$x \| y = x \lfloor\!\lfloor y + y \lfloor\!\lfloor x$	M1
$a \lfloor\!\lfloor x = a.x$	M2
$ax \lfloor\!\lfloor y = a(x \| y)$	M3
$(x + y) \lfloor\!\lfloor z = x \lfloor\!\lfloor z + y \lfloor\!\lfloor z$	M4

Here M2,M3 are axiom schemes where 'a' ranges over all constants (atoms) in A.

A_ω is the initial algebra of the equational specification (Σ_A, PA). It can be shown that each closed term over Σ_A is equal to a term not containing $||$ and $|\!\lfloor$ on basis of PA, moreover two terms built from A,+,• only are equal in PA iff they are equal on basis of A1-5 only. Thus, A_ω is an enrichment of the initial algebra of A1-5.

On A_ω an operator $(\ )_n$ is inductively defined for each $n \geq 1$ by

$$(a)_n = a$$

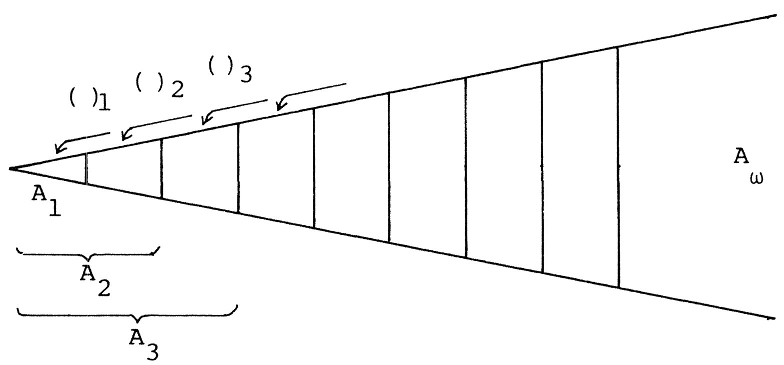
$$(ax)_{n+1} = a(x)_n$$

$$(ax)_1 = a$$

$$(x + y)_n = (x)_n + (y)_n \ .$$

Note that $(x)_n \in A_\omega$ again.

Let $p \equiv_n q$ in A_ω if $(p)_n = (q)_n$. Clearly, $\equiv_n$ is a congruence on A_ω. We write A_n for $A_\omega/\equiv_n$. Note that $(\ )_{n+1}$ maps A_{n+1} onto A_n.



A^∞ finally is the projective limit of the system

$$A_1 \xleftarrow[(\ )_1]{} A_2 \xleftarrow[(\ )_2]{} A_3 \xleftarrow[(\ )_3]{} \cdots A_n \xleftarrow[(\ )_n]{} A_{n+1} \cdots$$

The cardinality of A^∞ is $2^{\aleph_0}$, the cardinality of the continuum, even if $|A| = 1$.

2. ALGEBRAIC SPECIFICATION OF STATE TRANSITION SYSTEMS

A *state transition system* is a set V together with a set A of transition labels and a set $R \subseteq V \times A \times V$ of transitions.

In the sequel we will assume V to be finite.

If $v_1, v_2 \in V$ and $(v_1, a, v_2) \in R$ one writes $v_1 \xrightarrow{a} v_2$ or a: $v_1 \longrightarrow v_2$, both indicating the existence of a transition from v_1 to v_2 labeled by a.

An *algebraic specification* of a transition system is a quadruple

$$<(\Sigma,E),(A,T)>.$$

Here (Σ,E) is an equational specification. Its initial algebra semantics $I(\Sigma,E)$ is some (many-sorted) algebra. Its domain will act as the state set V. Further, A is a (finite) set of *action names (labels)* and T is a finite set of *transition rules*. Each transition rule has the following form:

$$a:\ t_1(X) \longrightarrow t_2(X).$$

Here X is a list of variables for Σ, and $t_1(X), t_2(X)$ are terms over $\Sigma(X)$, and $a \in A$.

The *state transition system* $\mathcal{M}<(\Sigma,E),(A,T)>$ *specified by* $<(\Sigma,E),(A,T)>$ has the domain of $I(\Sigma,E)$ as state space, A as action name set and the set R of transitions contains all triples (v_1, a, v_2) such that for some transition rule a: $t_1(X) \longrightarrow t_2(X)$ in T and some valuation σ: $X \longrightarrow I(\Sigma,E)$ of the free variables, in $I(\Sigma,E)$,

$$val_\sigma(t_1(X)) = v_1 \text{ and } val_\sigma(t_2(X)) = v_2.$$

REMARK. Using algebraic specifications many transition systems can be specified. A substantial classification and structure theory is conceivable here. However, we will only point out the fundamental restriction that if the transition system (V,A,R) is to have an algebraic specification (up to isomorphism, of course) then V must be finite or countable.

3. BEHAVIOURAL ABSTRACTION

Let (V,A,R) be a state transition system. Let V* denote the set of states in V which have the property that for some 'a' a transition a: s $\longrightarrow$ s' exists.

To each $s \in V^*$ a process $\pi_{(V,A,R)}(s) \in A^\infty$ is assigned. (We will usually omit the subscript.) The process $\pi(s)$ embodies the behaviour of (V,A,R) with s as initial state.

Using a simultaneous induction the $(\pi(s))_n$ are defined for all $s \in V^*$ (simultaneously), as follows:

$$(\pi(s))_1 = \sum\{a \mid a \in A, \exists s' \; a: s \longrightarrow s'\}$$

$$(\pi(s))_{n+1} = \sum \{a \mid a \in A, \; \exists s' \in V - V^* \; a: s \longrightarrow s'\} \; +$$

$$\sum \{a(\pi(s'))_n \mid a \in A, \; a: s \longrightarrow s', \; s' \in V^*\} \; .$$

Note that these sums are computed within the structures A_n. As A_n is finite, infinite sums are in fact just finite sums; this justifies the $\sum$-notation.

Thus for $s \in V^*$:

$$\pi(s) = \lim (\pi(s))_n \; .$$

Interestingly, a universal transition system can be easily manufactured from A^∞. Indeed, let $V_A = A^\infty \cup \{o\}$, where o is some new object, and define R_A as follows (for each $a \in A$):

$$a: \; a \longrightarrow o$$

$$a: \; ap \longrightarrow p$$

$$a: \; ap + q \longrightarrow p \; .$$

Then (V_A, A, R_A) is a transition system with $V_A^* = A^\infty$ and $\pi(p) = p$ for $p \in V^*$.

If there is a transition path in (V_A, A, R_A) leading from p to q then q is called a *subprocess* of p. Sub(p) denotes the class of subprocesses of p. Now

$$(\mathrm{Sub}(p), A, R \cap \mathrm{Sub}(p) \times A \times \mathrm{Sub}(p))$$

is a sub-transition-system of (V_A, A, R_A) which in a sense is the smallest system still containing p.

REMARK. Let $p = \lim_n \sum_{\alpha \in \{a,b\}^n} \alpha$ be the process mentioned in the introduction. Then p is the behaviour of a countable transition system but $|Sub(p)| = 2^{\aleph_0}$.

To see that Sub(p) has the cardinality of the continuum , note that for each $g: \omega \longrightarrow \{a,b\}$ the process $p_g = \lim_n g(0) \ldots g(n)$ is in Sub(p).

On the other hand, consider the following transition system. Let γ_n', $n \in \omega$ be an enumeration of all functions $g:\omega \longrightarrow \{a,b\}$ which have on all but finitely many arguments value a.

Now let: $V = \omega \times \omega \cup \{0\}$

$$A = \{a,b\}$$

$$R = \{0 \xrightarrow{a} (n,0)\ ,\ 0 \xrightarrow{b} (n,0) \mid n \in \omega\}$$

$$\cup \{(n,i) \xrightarrow{g_n(i)} (n,i+1) \mid n,i \in \omega\}$$

then $\pi_{(V,A,R)}(0) = p$.

4. EXAMPLES OF PROCESS SPECIFICATION

We will describe three simple examples of process specifications. First of all we need a precise definition.

DEFINITION. Let $p \in A^\infty$. An *algebraic specification* of p is a state space specification $<(\Sigma,E),(A,T)>$ and a closed term t such that

$$\pi_{\mathcal{M}<(\Sigma,E),(A,T)>}(t) = p .$$

REMARK. When $<(\Sigma,E),(A,T)>$ is given it also yields notations for parameterized families of processes. In general $\pi(t(x))$ is a process iff $t(x) \in V^*$.

The terms t(x) such that for all x, $t(x) \in V^*$ form a Π^0_2 collection of terms. For each closed term $t \in V^*$ a process notation $\pi(t)$ is given for a process in A^∞.

From the point of view of process algebra one is interested in identities that are true of the $\pi(t)$ in $(A^\infty, +, \cdot, \|, \|\!\_)$.

EXAMPLE 1. A counter: $\Sigma = ((\omega,S,0),\emptyset)$

$E = \emptyset$

A = SUCC, PRED, NULL

T = SUCC: $X \longrightarrow S(X)$
PRED: $S(X) \longrightarrow X$
NULL: $0 \longrightarrow 0$

Now $\pi(0)$ represents a counter in A^{∞}.

EXAMPLE 2. A stack with stack alphabet $\{a,b\} = B$:

Σ = sorts ST, B
constants $\emptyset \in ST$, $a,b \in B$
functions push: $B \times ST \longrightarrow ST$

$E = \emptyset$

A = PUSHa, PUSHb, TOPa, TOPb, $\emptyset$, EMPTY

T = EMPTY: $\emptyset \longrightarrow \emptyset$
PUSHa: $X \longrightarrow push(a,X)$
PUSHb: $X \longrightarrow push(b,X)$
TOPa: $push(a,X) \longrightarrow push(a,X)$
TOPb: $push(b,X) \longrightarrow push(b,X)$
POP: $push(u,X) \longrightarrow X$

Now $\pi(\emptyset)$ acts as a stack in A^{∞}.

EXAMPLE 3. A bag over the set $B = \{a_1,\ldots,a_k\}$.

Σ = sorts B, BAGS
constants $\emptyset \in BAGS$, $a_1,\ldots,a_k \in B$
functions INS: $B \times BAGS \longrightarrow BAGS$

$E = INS(x,INS(y,X)) = INS(y,INS(x,X))$

$A = \{a_1,\ldots,a_k,\underline{a}_1,\ldots,\underline{a}_k\}$

$T = a_i: X \longrightarrow INS(a_i,X)$ for $a_i \in A$

$\underline{a}_i: INS(a_i,X) \longrightarrow X$ for $a_i \in A$

Now $\pi(\emptyset)$ acts as a bag in A^{∞}. Here a_i means: put a_i in the bag and $\underline{a}_i$ means: get a_i from the bag.

In the case of the bag we are able to derive interesting mathematical identities. For instance:

(i) $\pi(INS(a,X)) = \underline{a} \parallel \pi(X)$

$$\text{(ii)} \quad \pi(\emptyset) = \sum_{i=1}^{k} a_i \cdot (\underline{a}_i \parallel \pi(\emptyset))$$

The second identity was already used as a definition of the bag in BERGSTRA & KLOP [9]. We can now formally validate the identity against the transition system specification which is already convincing on intuitive grounds.

Using induction on n one shows simultaneously for all sequences $a_{h1},\ldots,a_{hk}$ that

$$(*) \quad (\pi(\text{INS}(a_{h1},\ldots,a_{hk},\emptyset)))_n = (\underline{a}_{h1} \parallel \ldots \parallel \underline{a}_{hk} \parallel \pi(\emptyset))_n$$

Basis: $n = 1$. Then

$$(\pi(\text{INS}(\vec{a},\emptyset)))_1 = a_1 + \ldots + a_k + \underline{a}_{h1} + \ldots + \underline{a}_{hk} = (\underline{a}_{h1} \parallel \ldots \parallel \underline{a}_{hk} \parallel \pi(\emptyset))_1$$

Induction step: $n = m + 1$. Then

$$(\pi(\text{INS}(\vec{a},\emptyset)))_{m+1} =$$

$$= \sum_{i=1}^{k} a_i(\pi(\text{INS}(a_i,\text{INS}(\vec{a},\emptyset))))_m +$$

$$\sum_{i=1}^{k} \underline{a}_{hi}(\pi(\text{INS}(\underline{a}_{h1},\ldots,\underline{a}_{hi-1},\underline{a}_{hi+1},\ldots,\underline{a}_{hk},\emptyset))) =$$

$$= \sum_{i=1}^{k} a_i(\underline{a}_i \parallel \underline{a}_{h1} \parallel \ldots \parallel \underline{a}_{hk} \parallel \pi(\emptyset))_m +$$

$$\sum_{i=1}^{k} \underline{a}_{hi}(\underline{a}_{h1} \parallel \ldots \parallel \underline{a}_{hi-1} \parallel \underline{a}_{hi+1} \parallel \ldots \parallel \underline{a}_{hk} \parallel \pi(\emptyset)) =$$

$$= ((\sum_{i=1}^{k} a_i(\underline{a}_i \parallel \pi(\emptyset))_m) \mathbin{\lfloor\!\lfloor} (\underline{a}_{h1} \parallel \ldots \parallel \underline{a}_{hk}))_{m+1} +$$

$$(\underline{a}_{h1} \parallel \ldots \parallel \underline{a}_{hk}) \mathbin{\lfloor\!\lfloor} \pi(\emptyset) \qquad (**)$$

Now

$$(\underline{a}_i \parallel \pi(\emptyset))_m = (\pi(\text{INS}(a_i,\emptyset)))_m$$

by induction hypothesis and

$$\pi(\emptyset) = \sum_{i=1}^{k} a_i \; \pi(\text{INS}(a_i,\emptyset))$$

by definition of π. This yields

$$(\pi(\emptyset))_{m+1} = \sum_{i=1}^{k} a_i(\pi(\mathrm{INS}(a_i,\emptyset)))_m = \sum_{i=1}^{k} a_i(\underline{a}_i \parallel \pi(\emptyset))_m .$$

Substituting this identity in (**) we obtain:

$$\pi(\mathrm{INS}(\vec{a},\emptyset))_{m+1} =$$
$$(\pi(\emptyset) \mathbin{\lfloor\!\lfloor} (\underline{a}_{h1} \parallel \ldots \parallel \underline{a}_{hk}))_{m+1} + ((\underline{a}_{h1} \parallel \ldots \parallel \underline{a}_{hk}) \mathbin{\lfloor\!\lfloor} \pi(\emptyset))_{m+1} =$$
$$(\underline{a}_{h1} \parallel \ldots \parallel \underline{a}_{hk} \parallel \pi(\emptyset))_{m+1}$$

as required. Using (*) we find the second identity:

$$\pi(\emptyset) = \sum_{i=1}^{k} a_i\pi(\mathrm{INS}(a_i,\emptyset)) = \sum_{i=1}^{k} a_i(\underline{a}_i \parallel \pi(\emptyset)).$$

5. A COMPARISON OF THE RELATIVE POWER OF THE THREE SPECIFICATION MECHANISMS

Given an alphabet A we will consider three classes of processes in A^{∞}.

(1) K_{eff}: processes with a computable projective sequence.

(2) K_{rec}: processes that are recursively defined over $(A,+,\cdot,\parallel,\mathbin{\lfloor\!\lfloor})$ by means of a system of guarded equations.

(3) K: processes that can be specified via an equational specification of a state transition system.

The results of this section are all summarised in the following theorem:

THEOREM.

$$K_{rec} \subsetneq K_{eff} \neq K$$

PROOF. We will first show that $K \neq K_{eff}$, to this end we start with defining computable transition systems.

DEFINITION. A transition system (V,A,R) is *computable* if there exists a mapping $\phi: \omega \to V$ such that the relation $R_{\phi} \subseteq \omega \times A \times \omega$ defined by

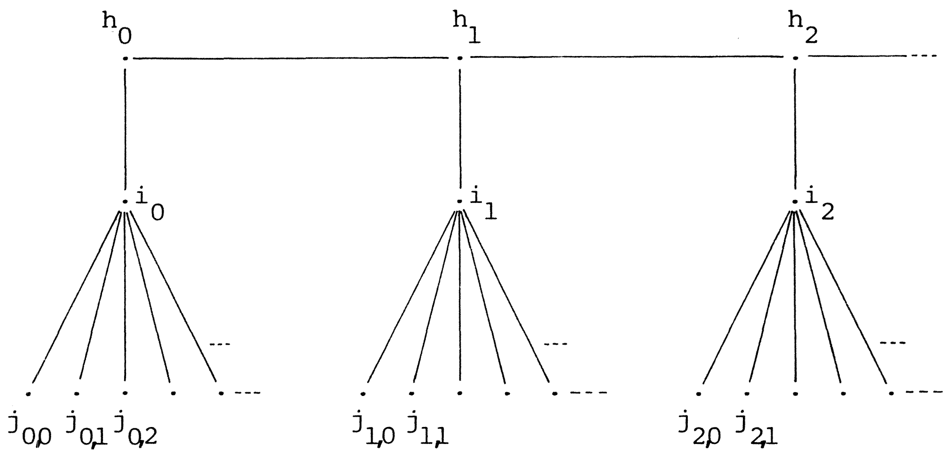
$$R_{\phi}(n,a,m) \iff R(\phi(n),a,\phi(m))$$

is decidable.

PROPOSITION. *A computable transition system can be algebraically specified.*

PROOF. This is a straightforward corollary of algebraic specification theory (see BERGSTRA & TUCKER [9]). □

Next we introduce a transition system (V,A,R_Z). Let the state space V be depicted as follows.



Let $Z \subseteq \omega$ be some recursively enumerable but not recursive relation. There is a computable relation $\bar{Z} \subseteq \omega$ such that for all n

$$Z(n) \Longleftrightarrow \exists m\ \bar{Z}(n,m).$$

Now a transition relation R_Z is defined on V, with A = {a,b}, as follows:

$$\text{a: } h_\ell \longrightarrow h_{\ell+1} \quad (\ell \in \omega)$$

$$\text{b: } h_\ell \longrightarrow i_\ell \quad (\ell \in \omega)$$

$$\text{c: } i_\ell \longrightarrow j_{\ell,m} \quad (\ell,m \in \omega \text{ and } \bar{Z}(\ell,m))$$

The transition system (V,A,R_Z) thus obtained is evidently computable. Hence, according to the above Proposition, (V,A,R_Z) has an algebraic specification and so $\pi(h_0) \in K$. We will show that $\pi(h_0) \notin K_{eff}$, whence $K \not\subseteq K_{eff}$.

Obviously, $(\pi(h_0))_{n+2}$ contains an atom c at depth n+2 if and only if $n \in Z$. Therefore Z is computationally reducible to the sequence $(\pi(h_0))_n$, which consequently cannot be computable. That is: $\pi(h_0) \notin K_{eff}$.

To prove that K_{eff} contains K it suffices to note that working modulo n a system of guarded equations posesses a unique solution which uniformly depends on n.

That K_{eff} and K_{rec} are unequal follows from an example given in BERGSTRA & KLOP [8]. There it is shown that an infinite recursively defined process must have an infinite regular trace. Thus for instance the process babaabaaabaaaabaaaaab..... is not rcursively defined.

Furthermore in[8] we show that recursively defined processes are finitely branching and that the corresponding tree can be effectively generated.This implies that each process in K_{rec} is the behaviour of a computable transition system, which in turn has an algebraic specification. So we conclude that K_{rec} is a subset of K.

The relation between K_{eff} and K is still unclear to us. In ROUNDS[16] it is shown that each process over a finite A is the behaviour of some countable transition system. The proof rests on an elegant application of the compactness theorem for first order logic. It is easily seen that as a corollary to this proof each process in K_{eff} is the behaviour of a transition system which is computable relative to the Halting problem. The problem is, whetheror not such transition systems can be simulated by means of semi-computable ones which admit an algebraic specification. For this issue it might be interesting to consider final algebra semantics as well as a specification principle for transition systems.

REFERENCES

[1] ARNOLD, A.,
Sémantique des processus communicants,
R.A.I.R.O. Informatique théorique/Theoretical Informatics, Vol.15, No.2,p.103-139 (1981).

[2] BACK, R.J.R. & H. MANNILA,
A refinement of Kahn's semantics to handle non-determinism and communication,
Proc. of the ACM SIGACT-SIGOPS Symp. on Principles of Distributed Computing, Ottawa, Canada, August 1982, p.111-120.

[3] DE BAKKER, J.W. & J.I. ZUCKER,
Denotational semantics of concurrency,
Proc. 14th ACM Symp. on Theory of Computing, p.153-158 (1982).

[4] DE BAKKER, J.W. & J.I. ZUCKER,
Processes and the denotational semantics of concurrency,
Department of Computer Science Technical Report IW 209/82,
Mathematisch Centrum, Amsterdam 1982.

[5] BERGSTRA, J.A. & J.W. KLOP,
Fixed point semantics in process algebras,
Department of Computer Science Technical Report IW 206/82,
Mathematisch Centrum, Amsterdam 1982.

[6] BERGSTRA, J.A. & J.W. KLOP,
Process algebra for communication and mutual exclusion,
Department of Computer Science Technical Report IW 218/83,
Mathematisch Centrum, Amsterdam 1983.

[7] BERGSTRA, J.A. & J.W. KLOP,
A process algebra for the operational semantics of static data flow networks,
Department of Computer Science Technical Report IW 222/83,
Mathematisch Centrum, Amsterdam 1983.

[8] BERGSTRA, J.A. & J.W. KLOP,
The algebra of recursively defined processes and the algebra of regular processes,
Department of Computer Science Technical Report IW 2../83,
Mathematisch Centrum, Amsterdam 1983.

[9] BERGSTRA, J.A. & J.V. TUCKER,
Initial and final algebra specifications, two characterisation theorems,
SIAM J. Comp. Vol.12,2(1983),p.366-387.

[10] BROCK, J.D. & W.B. ACKERMAN,
Scenarios: a model of non-determinate computation,
Proc. Formalization of Programming concepts (J. Díaz & I. Ramos, eds.),p.252-259, Springer LNCS 107, 1981.

[11] HOARE, C.A.R.,
Communicating sequential processes,
CACM 21 (1978),p.666-677.

[12] MILNER, R.,
A Calculus for Communicating Systems,
Springer LNCS 92, 1980.

[13] PARK, D.M.R.,
Concurrency and automata on finite sequences,
Computer Science Department, University of Warwick (1981).

[14] PRATT, V.R.,
On the composition of processes,
Proc. 9th ACM Symp. on Principles of Programming Languages, p.213-223, 1982.

[15] REM, M.,
Partially ordered computations, with applications to VLSI design,
Proc. of the 4th Advanced Course on Foundations of Computer Science, Part 2, Mathematical Centre Tracts 159, Mathematisch Centrum 1983.

[16] ROUNDS, W.C.,
On the relationships between Scott domains, synchronisation trees and metric spaces,
University of Michigan 1983.

69F11
69F12
69F32
69F43

ONTVANGEN 2 2 AUG. 1983