Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

A. Ponse

Process algebra and Hoare's logic

Computer Science/Department of Software Technology Note CS-N8802 December

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

69D41, 69F31, 69F32

Process Algebra and Hoare's Logic

Alban Ponse
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

Abstract

A Hoare-like logic is introduced for deriving 'partial correctness assertions' of the form $\{\alpha\}\, p\, \{\beta\}$, where α, β are unary predicates over some state space S and p is an expression over a recursive, non-uniform language containing global nondeterminism $(+)$ and sequential composition $(\cdot)$. This logic is (relatively) complete if only guarded recursion is considered.

Key Words & Phrases: process algebra, side-effects, Hoare's logic, partial correctness assertions.
1985 Mathematical Subject Classification: 68Q55, 68Q60.
1980 Mathematical Subject Classification: 68B10, 68F20.
1982 CR Categories: D.3.1, F.3.1, F.3.2.
Note: The author received full support from the European Communities under ESPRIT project no. 432, An Integrated Formal Approach to Industrial Software Development (METEOR).

1 Introduction

We present an application of Hoare's logic, an axiomatic method for proving programs correct, in the field of process algebra. For a survey of Hoare's logic, see [1]. Process algebra, an algebraic approach to the study of (concurrent) processes, provides us with a language for the specification of processes and with axiom systems concerning the equality relation and the operators involved in such specifications. We will concentrate on the sequential fragment of ACP, the Algebra of Communicating Processes (ACP is discussed in eg. [5]).

We consider the use of a state space S on which atomic actions have side effects and we define semantical notions based on transition rules in the style of Plotkin, in which these side effects are inserted (cf. [9]). Thus we consider a *non-uniform* process language (this notion is discussed in eg. [4]), that is, we regard processes as state transformers in the sense that an execution of a process is related to some *initial state* and, when terminating successfully, to a *final state*. We specify this behaviour by means of 'partial correctness assertions'

$$\{\alpha\}\, p\, \{\beta\}$$

(so called 'Hoare triples') where α, β are unary predicates over S and p represents a process. The interpretation of such a partial correctness assertion is as follows:

> If the execution of a process p, starting from an initial state satisfying α terminates, then the resulting final state satisfies β.

The logic we present contains a proof system H in which partial correctness assertions are *derivable*.

The main result of this paper concerns the *relative completeness* of H: Considering all true assertions about a state space S as axioms in derivations, we show that derivability and truth of partial correctness assertions concerning *guardedly specifiable* processes coincide. In order to prove this result we

Sort :	P		the set of processes
Constants:	a	$\in P$	for any atomic action $a \in A$
	δ	$\in P$	deadlock ($\delta \notin A$)
	τ	$\in P$	silent action ($\tau \notin A$)
	$<x \mid E>$	$\in P$	a process specified by a guarded system E
Functions :	$+$:	$P \times P \to P$	alternative composition (sum)
	$\cdot$:	$P \times P \to P$	sequential composition (product)
	π_n :	$P \to P$	projection; $n \in \mathbb{N}$
Predicates:	B_n	$\subseteq P$	boundedness up to level n

Table 1: The signature Σ

need the Approximation Induction Principle (and some axioms going with it), a common principle in process algebra concerning the equality relation over process expressions. So we consider partial correctness assertions *modulo* a set of axioms. We also present a subsystem of H for which we prove the relative completeness of partial correctness assertions concerning *linear specifiable* processes, without using process algebra axioms.

This paper is organized as follows: Section 1 is concluded with a short introduction to process algebra and trace theory. In section 2 we discuss how to combine process algebra and Hoare's logic by defining a semantical concept of processes having side effects and formally introducing partial correctness assertions. In order to relate our semantical notions with more common semantics of process algebra, we provide a 'Semantical tour' in section 3. In sections 4-7 we show that the proof system H is (relatively) complete. We proceed in four stages (one per section): Starting with partial correctness assertions about 'finite processes', we develop three intermediate proof systems dealing with consecutively larger classes of process expressions. By showing that all these systems are complete, we can prove the completeness of H for the class of processes 'guardedly specifiable over $\mathrm{BPA}_{\delta\tau}$ + REC'. Finally, in section 8 we shortly discuss some problems of partial correctness assertions concerning processes specified over ACP_τ, a common axiom system in process algebra.

1.1 Process algebra, definitions and intuitions

In this paper a central role is played by the axioms listed in table 2. A parameter with respect to these axioms is a set A of atomic actions. For each atomic action $a \in A$ there is a constant a in the language, representing the process, starting with an a-step and terminating after some time. There is a special constant δ, representing deadlock, i.e. the acknowledgement of a process that it cannot do anything any more. Also we have a constant τ, representing silent action (unobservable or invisible activity), i.e. an internal machine step that cannot be observed. Furthermore there are constants $<x \mid E>$, representing processes specifiable by a 'guarded system' (these will be discussed in a moment). In table 1 we give the signature Σ of the axiomatic theory relevant to this paper. Let $x, y, z, \ldots$ be syntactic variables for process expressions (i.e. the set of terms over Σ) and let P denote the set of closed process expressions over Σ with $p, q, r, \ldots$ as syntactic variables. In a product $x \cdot y$ the symbol $\cdot$ is often omitted, we take $\cdot$ to be most binding of all operators and $+$ to be least binding (eg. $xx' + yy' = (x \cdot x') + (y \cdot y')$). We provide some intuitions concerning the axioms in table 2 where A_δ is short for $A \cup \{\delta\}$ and $a \in A_\delta$:

$\mathrm{BPA}_{\delta\tau}$ (*Basic Process Algebra* with *deadlock* and *abstraction*) The binary operator $+$ represents 'alternative composition' (sum): $x + y$ represents the process which first makes a choice between its summands x and y, and then proceeds with the chosen course of action. There is no order in

BPA$_{\delta\tau}$				
$(a \in A_\delta)$	$x+y=y+x$	A1	$x+\delta=x$	A6
	$x+(y+z)=(x+y)+z$	A2	$\delta x=\delta$	A7
	$x+x=x$	A3	$x\tau=x$	T1
	$(x+y)z=xz+yz$	A4	$\tau x+x=\tau x$	T2
	$(xy)z=x(yz)$	A5	$a(\tau x+y)=a(\tau x+y)+ax$	T3
REC	$<x \mid E> = <t_x \mid E>$			
PR				
$(a \in A_\delta)$	$\pi_n(\tau)=\tau$	PR1	$\pi_n(\tau x)=\tau\cdot\pi_n(x)$	PR4
	$\pi_0(ax)=\delta$	PR2	$\pi_n(x+y)=\pi_n(x)+\pi_n(y)$	PR5
	$\pi_{n+1}(ax)=a\cdot\pi_n(x)$	PR3		
B				
$(a \in A_\delta)$	$B_0(x)$	B1	$\dfrac{B_n(x)}{B_{n+1}(ax)}$	B4
	$B_n(\tau)$	B2		
	$\dfrac{B_n(x)}{B_n(\tau x)}$	B3	$\dfrac{B_n(x) \quad B_n(y)}{B_n(x+y)}$	B5
AIP$^-$	$\dfrac{\forall n \in \mathbb{N}(\pi_n(x)=\pi_n(y)\ ,\ B_n(x))}{x=y}$			

Table 2: The axioms of BPA$_{\delta\tau}$ + REC + PR + B + AIP$^-$

the alternatives (axiom A1) and a sequence of choices can be regarded as a single choice between all alternatives (axiom A2). A choice between two identical alternatives is neglected (axiom A3) and δ is never chosen in the presence of an alternative (axiom A6). The latter implies that δ does *not* represent deadlock if it is occurring in a sum context offering alternatives. The binary operator $\cdot$ represents 'sequential composition' (product): $x \cdot y$ (or xy, for short) represents the process x, followed after possible termination of x by y. The process x does not terminate if it ends in deadlock (axiom A7), or if it performs an infinite sequence of (possibly invisible) actions. The axiom A5 speaks for itself.

The axiom A4 describes the interaction between alternative and sequential composition. Because there is no axiom $x(y+z) = xy+xz$ we can discern the 'moment' of choice: Typically the process $a\delta + aa$ may deadlock, in contradistinction to the process $a(\delta + a)$, which equals the process aa, according to the axiom A6.

A possible intuition concerning the constant τ, supported by the axioms T1-T3 (the τ-laws of Milner), is to regard τ as representing invisible activity of a machine, executing processes such that only the beginning of atomic actions is visible and the execution of any atomic action takes a finite amount of time. The axiom T1 states that such invisible activity always takes a finite amount of time and the axiom T2 tells us that τ may take no time at all. The axiom T3, expressing that a process specified by $a(\tau x + y)$ has a summand ax, is derivable if $a \equiv \delta$. Otherwise, if $a \in A$, it reflects the possibility that some time after the start of an atomic action a is observed, this process can be in a state where it can only proceed with x and not any more with the process y.

REC (*Recursion*) Here E is a *recursive specification*, i.e. a set of equations $E = \{x = t_x \mid x \in V_E\}$ where V_E is a set of variables and t_x some process expression over Σ only containing variables of V_E (the set V_E need not be finite). A *solution* of E is an interpretation of the variables in V_E as processes in a certain semantics, such that the equations of E are satisfied. We now introduce a syntactical restriction on recursive specifications such that for all $x \in V_E$ there is exactly one solution of E in our intended semantics. A recursive specification $E = \{x = t_x \mid x \in V_E\}$ is called *guarded* if each occurrence of a variable x in the expressions t_x occurs in a subterm aM with $a \in A$. We speak of *guarded systems* instead of guarded recursive specifications. For all guarded systems E not containing π_n-operators there are constants $<x \mid E>$ for $x \in V_E$ in Σ, denoting the x-component of a solution of E.

Let $E = \{x = t_x \mid x \in V_E\}$ be a guarded system, and t a process expression. Then $<t \mid E>$ denotes the process expression in which each occurrence of $x \in V_E$ in t is replaced by $<x \mid E>$. Now the axiom REC expresses the fact that $<x \mid E>$ is a solution of E.

If we assume that the variables in a recursive specification are chosen freshly, there is no need to repeat E in each occurrence of $<x \mid E>$. Variables reserved in this way are called *formal variables* and denoted by capital letters. We adopt the convention that $<X \mid E>$ can be abbreviated by X once E is declared. As an example consider $E = \{X = aX\}$: Now a statement $X = aaX$ abbreviates $<X \mid \{X = aX\}> = aa <X \mid \{X = aX\}>$.

PR (*Projection*) The projection operator π_n ($n \in \mathbb{N}$) stops a process after it has performed n atomic actions[1]. If $\pi_n(x) = x$ for some n we will call x *finite*. More specifically $\pi_n(x)$ is of *depth* n if $\pi_n(x) = x$ and for all $k < n(\pi_k(x) \neq x)$.

B (*Bounded nondeterminism*) The predicate B_n holds if the nondeterminism displayed by a process before its n^{th} atomic action is bounded. When only considering guarded recursion the axioms of B are always satisfied for closed process expressions. These axioms are used in:

[1] Note that we use the version of PR in which 'π_0' is defined.

AIP$^-$ (*Weak Approximation Induction Principle*) This principle states that every process which has a guarded specification is determined by its finite projections. This 'weak' version of the Approximation Induction Principle holds in any semantical structure in which we will interpret partial correctness assertions. It appears to be crucial in the completeness result we prove. See [6] for this principle.

1.2 About trace theory

We first give an enumeration of some general definitions and notations:

- For any alphabet $\mathcal{A}$ we use $\mathcal{A}^*$ to denote the set of finite sequences over $\mathcal{A}$. We write λ for the empty sequence and a for the sequence consisting of the single symbol $a \in \mathcal{A}$. If $\sigma, \sigma' \in \mathcal{A}^*$, then $\sigma * \sigma'$, often abbreviated as $\sigma\sigma'$, denotes the concatenation of the sequences σ and σ'.
- By $\#\sigma$ we denote the *length* of a sequence $\sigma \in \mathcal{A}^*$: $\#\lambda \overset{\text{def}}{=} 0$; if $a \in \mathcal{A}$, then $\#a \overset{\text{def}}{=} 1$; $\#\sigma\sigma' \overset{\text{def}}{=} \#\sigma + \#\sigma'$.
- If $V \subseteq \mathcal{A}^*$, then $\Pi_n(V)$ denotes the subset of V containing all elements of length not exceeding n, thus $\Pi_n(V) \overset{\text{def}}{=} \{\sigma \in V \mid \#\sigma \leq n\}$.
- If $V, W \subseteq \mathcal{A}^*$ and $\sigma \in \mathcal{A}^*$, then $\sigma * V$ (or σV) denotes the set $\{\sigma\rho \mid \rho \in V\}$ and $V * W$ (or VW) denotes the set $\bigcup\limits_{\sigma \in V} \sigma W$.

A *trace* of a process is a finite sequence that gives a possible order in which atomic actions can be performed by that process. Let $tr(x)$ denote the set of all traces of a process x. For each $x \in P$ we now introduce the set of *complete traces* of x, that is the subset of $tr(x)$ containing only the traces representing successful termination of x. Because we are (in the context of Hoare's logic) specifically interested in relating initial and (possible) final states of the execution of processes, we introduce an operator tr_c which returns the set of complete traces of a process:

Definition 1.2.1 *For all $x \in P$ let $tr_c(x) \subseteq A^*$ be the set of all complete traces of x:*

- $tr_c(\delta) \overset{\text{def}}{=} \emptyset$
- $tr_c(\tau) \overset{\text{def}}{=} \{\lambda\}$
- $tr_c(a) \overset{\text{def}}{=} \{a\}$ $(a \in A)$
- $tr_c(xy) \overset{\text{def}}{=} tr_c(x) * tr_c(y)$
- $tr_c(x + y) \overset{\text{def}}{=} tr_c(x) \cup tr_c(y)$

Using the axioms of table 2 it can be easily proved that $tr_c(\pi_n(x)) = \Pi_n(tr_c(x))$ and (if E is a guarded system) $tr_c(<x|E>) = \bigcup\limits_n tr_c(\pi_n(<x|E>))$. We may now define 'complete trace semantics':

Definition 1.2.2 *Two closed process expressions p and q are* complete trace equivalent,

$$p =_{ctr} q$$

if $tr_c(p) = tr_c(q)$.

We state without proof that the relation $=_{ctr}$ can be axiomatized by adding CTR to $\mathrm{BPA}_{\delta\tau} + \mathrm{REC} + \mathrm{PR} + \mathrm{B} + \mathrm{AIP}^-$, where CTR is the axiom system defined in table 3.

$x\delta = \delta$	CTR1
$\tau x = x$	CTR2
$x(y+z) = xy + xz$	CTR3

Table 3: CTR axioms

2 Combining process algebra and Hoare's logic

In this section we introduce processes having side effects in a formal way by defining functions *action* and *effect*, relating atomic actions with a state space. An operational semantics in which closed process expressions are identified if they perform the 'same behaviour' when started in the same initial state is discussed. Next we introduce partial correctness assertions, relating a process and a set of initial states to a set of possible final states, denoting successful termination. Furthermore a 'partial correctness semantics', defined by identifying closed process expressions if they satisfy the same partial correctness assertions is presented. Finally we introduce proof systems, by which we can derive partial correctness assertions.

2.1 Processes as state transformers

We regard processes as having a state. Let S be a nonempty set of states, with typical elements $s, s', \ldots$. The idea is that the execution of an action a in state s results in an action a' representing the activity of this execution (an atomic action, τ or δ), and in a resulting state s' [2]. This idea is formalized by *given* (total) functions

$$\mathit{action} : A_{\delta\tau} \times S \rightarrow A_{\delta\tau} \quad \text{and} \quad \mathit{effect} : S \times A_{\delta\tau} \rightarrow S$$

which determine the relation between elements a of $A_{\delta\tau}$ and elements s of S, the set of states. With $\mathit{action}(a, s)$ we denote the activity which represents the execution of a in state s; with $\mathit{effect}(s, a)$ we denote the resulting state. It is assumed that

1. $\forall s \in S(\mathit{action}(\tau, s) = \tau)$ and $\forall s \in S(\mathit{action}(\delta, s) = \delta)$

for, in each state an invisible action must remain invisible and δ should indeed denote deadlock;

2. $\forall s \in S(\mathit{effect}(s, \tau) = s)$ and $\forall s \in S(\mathit{effect}(s, \delta) = s)$

because neither a τ-step, nor deadlock should alter a state. We demand that for all functions *action* and *effect* considered the properties 1 and 2 hold. Another way to state this is to demand that τ and δ are *inert* (with respect to the functions *action* and *effect*). We use the following abbreviations: $a(s)$ for $\mathit{action}(a, s)$; $s(a)$ for $\mathit{effect}(s, a)$. The functions *action* and *effect* were introduced in [2].

This 'operational view' of the execution of elements of A_τ in some state will be generalised to an operational semantics of closed process expressions based on transition rules in the style of Plotkin. We consider 'state labelled process expressions' (x, s) denoting the process x in state s and introduce for all $a \in A_\tau$ a binary transition relation $\xrightarrow{a}$ over state labelled process expressions. By the transition $(x, s) \xrightarrow{a} (y, s')$ we mean that by performing an action a the process x in state s can evolve into y

[2] As an example think of the representation of a program in a high level language as *Pascal* in process algebra. If a variable x is declared as an integer; an assignment $x := x + 1$ is regarded as an atomic action a and S denotes the set of valuations from declared variables to their full domains, then $a' = \tau$ if $s(x) <$ MAXINT and δ otherwise. Of course $s' = s[s(x) + 1/x]$.

in state s'. To represent successful termination we introduce a special element $\surd$ not in Σ and for all $a \in A_\tau$ a relation $\xrightarrow{a} (\surd, .)$ defined on $(P \times S) \times S$. Now the expression $(x, s) \xrightarrow{a} (\surd, s')$ denotes that the process x in state s can terminate successfully by performing a. In table 4 we present a proof system, the *effect rules*, by which we can derive transitions. Concerning the π_n-rules, note that any derivable transition $(p, s) \xrightarrow{c} (q, s')$ must have the form $(p, s) \xrightarrow{a(s)} (q, s')$ for some $a \in A_\tau$, since the 'basic' transitions are defined in this way.

We define $\overset{\sigma}{\twoheadrightarrow}$ for $\sigma \in (A_\tau)^*$ as the reflexive and transitive closure of the ternary relations $\xrightarrow{a}$ and $\xrightarrow{a} (\surd, .)$:

- $\dfrac{(x, s) \xrightarrow{a} (y, s')}{(x, s) \overset{a}{\twoheadrightarrow} (y, s')} \quad \dfrac{(x, s) \xrightarrow{a} (\surd, s')}{(x, s) \overset{a}{\twoheadrightarrow} (\surd, s')} \quad (a \in A_\tau)$
- $(x, s) \overset{\lambda}{\twoheadrightarrow} (x, s)$
- $\dfrac{(x, s) \overset{\sigma}{\twoheadrightarrow} (y, s') \quad (y, s') \overset{\rho}{\twoheadrightarrow} (z, s'')}{(x, s) \overset{\sigma\rho}{\twoheadrightarrow} (z, s'')} \quad \dfrac{(x, s) \overset{\sigma}{\twoheadrightarrow} (y, s') \quad (y, s') \overset{\rho}{\twoheadrightarrow} (\surd, s'')}{(x, s) \overset{\sigma\rho}{\twoheadrightarrow} (\surd, s'')}$

Instances of this relation will be called *effect reductions*.

We present two technical results concerning effect reductions and a standard operation on strings $\sigma \in (A_{\delta\tau})^*$:

- Let $\sigma \in (A_{\delta\tau})^*$, the string $\tilde{\sigma} \in (A_\delta)^*$ is obtained by replacing all τ-occurrences in σ with λ.

1. If $(p, s) \overset{\sigma_1}{\twoheadrightarrow} (\surd, s')$ and $(q, s') \overset{\sigma_2}{\twoheadrightarrow} (\surd, s'')$, then $(pq, s) \overset{\sigma_1\sigma_2}{\twoheadrightarrow} (\surd, s'')$.
2. If $(pq, s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')$, then there are $s'' \in S$, $\sigma_1, \sigma_2 \in (A_\tau)^*$ such that $(p, s) \overset{\sigma_1}{\twoheadrightarrow} (\surd, s'')$ $(q, s'') \overset{\sigma_2}{\twoheadrightarrow} (\surd, s')$ and $\widetilde{\sigma_1}\widetilde{\sigma_2} \equiv \tilde{\sigma}$.

Result 1 follows by a simple induction on $\#\sigma_1$, the length of σ_1, and for result 2 we need three intermediate results:

$$(i)\ (pq, s) \xrightarrow{a} (r, s') \implies \text{One of} \left\{ \begin{array}{ll} (p, s) \xrightarrow{a} (\surd, s''), & (q, s'') \xrightarrow{\tau} (r, s') \\ (p, s) \xrightarrow{\tau} (\surd, s''), & (q, s'') \xrightarrow{a} (r, s') \\ (p, s) \xrightarrow{a} (p', s'), & r \equiv p'q \\ (p, s) \xrightarrow{a} (\surd, s'), & r \equiv q \end{array} \right. \text{holds.}$$

$$(ii)\ (pq, s) \xrightarrow{a} (\surd, s') \implies \text{One of} \left\{ \begin{array}{ll} (p, s) \xrightarrow{a} (\surd, s''), & (q, s'') \xrightarrow{\tau} (\surd, s') \\ (p, s) \xrightarrow{\tau} (\surd, s''), & (q, s'') \xrightarrow{a} (\surd, s') \end{array} \right. \text{holds.}$$

$$(iii)\ (pq, s) \overset{\sigma}{\twoheadrightarrow} (r, s') \implies \text{One of} \left\{ \begin{array}{lll} (p, s) \overset{\sigma}{\twoheadrightarrow} (p', s'), & r \equiv p'q & \\ (p, s) \overset{\sigma_1}{\twoheadrightarrow} (\surd, s''), & (q, s'') \overset{\sigma_2}{\twoheadrightarrow} (r, s'), & \widetilde{\sigma_1}\widetilde{\sigma_2} \equiv \tilde{\sigma} \end{array} \right. \text{holds.}$$

We conclude this section with a definition concerning the syntax and semantics of state labelled closed process expressions and partial correctness assertions.

Definition 2.1.1 *A structure $\langle A, S, action, effect \rangle$ is a quadruple containing the set A of atomic actions, a nonempty set S of states and functions action $: A_{\delta\tau} \times S \to A_{\delta\tau}$ and effect $: S \times A_{\delta\tau} \to S$ such that δ and τ are inert.*

Remark that any structure $\mathcal{S}$ has a parameter A, the set of atomic actions. We use symbols $\mathcal{S}, \mathcal{S}'$ as syntactic variables for structures. Notice that any structure fixes the effect rules in table 4.

$a \in A_\tau$:	$(a, s) \xrightarrow{a(s)} (\surd, s(a))$	(if $a(s) \neq \delta$)	
$+$:	$\dfrac{(x, s) \xrightarrow{a} (x', s')}{(x + y, s) \xrightarrow{a} (x', s')}$	$\dfrac{(x, s) \xrightarrow{a} (\surd, s')}{(x + y, s) \xrightarrow{a} (\surd, s')}$	
	$\dfrac{(y, s) \xrightarrow{a} (y', s')}{(x + y, s) \xrightarrow{a} (y', s')}$	$\dfrac{(y, s) \xrightarrow{a} (\surd, s')}{(x + y, s) \xrightarrow{a} (\surd, s')}$	
$\cdot$:	$\dfrac{(x, s) \xrightarrow{a} (x', s')}{(xy, s) \xrightarrow{a} (x'y, s')}$	$\dfrac{(x, s) \xrightarrow{a} (\surd, s')}{(xy, s) \xrightarrow{a} (y, s')}$	
π_n :	$\dfrac{(x, s) \xrightarrow{a(s)} (y, s')}{(\pi_{n+1}(x), s) \xrightarrow{a(s)} (\pi_n(y), s')}$	$\dfrac{(x, s) \xrightarrow{a(s)} (\surd, s')}{(\pi_{n+1}(x), s) \xrightarrow{a(s)} (\surd, s')}$	(if $a \neq \tau$)
	$\dfrac{(x, s) \xrightarrow{\tau(s)} (y, s')}{(\pi_n(x), s) \xrightarrow{\tau(s)} (\pi_n(y), s')}$	$\dfrac{(x, s) \xrightarrow{\tau(s)} (\surd, s')}{(\pi_n(x), s) \xrightarrow{\tau(s)} (\surd, s')}$	
recursion :	$\dfrac{(<t_x \mid E>, s) \xrightarrow{a} (y, s')}{(<x \mid E>, s) \xrightarrow{a} (y, s')}$	$\dfrac{(<t_x \mid E>, s) \xrightarrow{a} (\surd, s')}{(<x \mid E>, s) \xrightarrow{a} (\surd, s')}$	
$\tau-$laws :	$(a, s) \xrightarrow{a(s)} (\tau, s(a))$	(if $a(s) \neq \delta$)	
	$\dfrac{(x, s) \xrightarrow{\tau} (y, s') \quad (y, s') \xrightarrow{a} (z, s'')}{(x, s) \xrightarrow{a} (z, s'')}$	$\dfrac{(x, s) \xrightarrow{\tau} (y, s') \quad (y, s') \xrightarrow{a} (\surd, s'')}{(x, s) \xrightarrow{a} (\surd, s'')}$	
	$\dfrac{(x, s) \xrightarrow{a} (y, s') \quad (y, s') \xrightarrow{\tau} (z, s'')}{(x, s) \xrightarrow{a} (z, s'')}$	$\dfrac{(x, s) \xrightarrow{a} (y, s') \quad (y, s') \xrightarrow{\tau} (\surd, s'')}{(x, s) \xrightarrow{a} (\surd, s'')}$	

Table 4: Effect rules

2.2 An operational semantics

Let $\mathcal{S}$ be some structure. The next step towards our operational semantics is to associate a ***transition system*** to any state labelled closed process expression (p, s), representing all possible transitions. The idea is that two process expressions p and q are operationally equivalent if they satisfy the following property: The representation of any execution of p in some initial state s (in terms of its performance of atomic actions) also represents an execution of q in initial state s, and vice versa. We now formalise this idea. Consider the graph $\mathcal{G}((p, s))$ defined as follows:

$$\begin{aligned} \text{NODES} &\overset{\text{def}}{=} \{(p', s') \,|\, \text{there is } \sigma \in (A_\tau)^* \text{ such that } (p, s) \overset{\sigma}{\twoheadrightarrow} (p', s')\} \cup \\ &\phantom{\overset{\text{def}}{=}} \{(\surd, s') \,|\, \text{there is } \sigma \in (A_\tau)^* \text{ such that } (p, s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')\} \\ \text{ARCS} &\overset{\text{def}}{=} \{k \xrightarrow{a} k' \,|\, k, k' \in \text{NODES and } k \xrightarrow{a} k' \text{ a transition}\} \end{aligned}$$

By defining the node (p, s) as the *root* of $\mathcal{G}((p, s))$, this construction yields $ts((p, s))$, the transition system associated to (p, s). Here the state s will be called the *initial* state of $ts((p, s))$. Any state s' such that $(\surd, s')$ is a node in $\mathcal{G}((p, s))$ will be called a ***final*** state of $ts((p, s))$. Now consider the set of all transition systems. In order to define an equality relation over this set, we use the notion of a bisimulation (see [8]):

Definition 2.2.1 *A binary relation $R \subseteq (\mathcal{P} \times S) \times (\mathcal{P} \times S)$ is a* bisimulation *if the following conditions are satisfied $(a \in A_\tau)$:*

1. *If $(p, s)R(q, s)$ and $(p, s) \xrightarrow{a} (p', s')$, then there is a (q', s') such that $(q, s) \xrightarrow{a} (q', s')$ and $(p', s')R(q', s')$.*
2. *If $(p, s)R(q, s)$ and $(q, s) \xrightarrow{a} (q', s')$, then there is a (p', s') such that $(p, s) \xrightarrow{a} (p', s')$ and $(p', s')R(q', s')$.*
3. *If $(p, s)R(q, s)$, then $(p, s) \xrightarrow{a} (\surd, s')$ for some s' if and only if $(q, s) \xrightarrow{a} (\surd, s')$ for some s'.*

Two transition systems $ts((p, s))$ and $ts((q, s))$ are bisimilar, *$ts((p, s)) \; \underline{\leftrightarrow} \; ts((q, s))$, if there exists a bisimulation R with $(p, s)R(q, s)$ (remark that equality of initial states is demanded here).*

The clauses 1 and 2 of this definition are called the *transfer property*. It is not difficult to see that $\underline{\leftrightarrow}$ is an equivalence relation. We now define an operational semantics:

Definition 2.2.2 *We will call two closed process expressions p and q* semantically equivalent *in $\mathcal{S}$,*

$$\mathcal{S} \models p =_{se} q$$

if for all $s \in S$ we have $ts((p, s)) \; \underline{\leftrightarrow} \; ts((q, s))$.

Remark that if we want to consider a structure $\mathcal{S} = \langle A, S, \mathit{action}, \mathit{effect} \rangle$ in which for two atomic actions a and b we have for all $s \in S$ that $a(s) = b(s)$ and $s(a) = s(b)$, then $\mathcal{S} \models a =_{se} b$. This reflects the circumstance that in $\mathcal{S}$ the constants a and b apparently denote the same atomic action. We finally state the following property of the relation $=_{se}$:

Theorem 2.2.3 *For all structures $\mathcal{S}$ the relation $=_{se}$ is a congruence with respect to the operators involved.*

Proof. In this proof we use a quite recent result: 'Bisimulation is a congruence if the transition rules satisfy a certain (liberal) syntactical format' (see [7]). Unfortunately the effect rules do not satisfy this format since they were defined independent of this result. We introduce related transition relations $\overset{a}{\rightsquigarrow}$ for $a \in A_\tau \cup A_\tau * \{\surd\}$ and a set of rules satisfying the format mentioned. We here just give a sketch of the rules:

- $(a,s) \overset{a(s)\surd}{\rightsquigarrow} (\delta, s(a))$ if $a(s) \neq \delta$
- $\dfrac{(x,s) \overset{a\surd}{\rightsquigarrow} (x',s')}{(xy,s) \overset{a}{\rightsquigarrow} (y,s')} a \in A_\tau \qquad \dfrac{(x,s) \overset{a}{\rightsquigarrow} (x',s')}{(xy,s) \overset{a}{\rightsquigarrow} (x'y,s')} a \in A_\tau$

for a related signature $\Sigma^{\rightsquigarrow}$ over $P \times S$ and abbreviating $(p,s) \overset{\rightsquigarrow}{\cdot} (q,s)$ as (pq,s). In this style we can represent all effect rules. It can be proved that

1. $(p,s) \xrightarrow{a} (\surd, s') \iff (p,s) \overset{a\surd}{\rightsquigarrow} (\delta, s')$
2. $(p,s) \xrightarrow{a} (q, s') \iff (p,s) \overset{a}{\rightsquigarrow} (q, s')$

Now $ts((p,s)) \leftrightarroweq ts((q,s))$ if and only if the 'related' transition systems are bisimilar. □

2.3 Assertions about processes

Our goal is to prove partial correctness assertions about processes, that is, given a structure $\mathcal{S} = \langle A, S, \mathit{action}, \mathit{effect}\rangle$, to relate a closed process expression p with subsets of S in the following way:

> Two subsets `initial`(S) and `final`(S) of S are related by p if for any execution of p, starting from some initial state in `initial`(S) and terminating successfully, the resulting final state is in `final`(S).

In order to reason formally we define a language $\mathcal{L}_A$ as follows:

variables:	$v_0, v_1, v_2, \ldots$	
unary predicate symbols:	$\mathit{stop}(a,.)$	(for all $a \in A_{\delta\tau}$)
unary function symbols:	$\mathit{effect}(.,a)$	(for all $a \in A_{\delta\tau}$)
connectives:	$\neg, \vee, \wedge, \rightarrow, \leftrightarrow$	
auxiliary symbols:	), , (	

The notation $\mathit{stop}(a,.)$ defines the place for the terms to be substituted: We write '$\mathit{stop}(a,t)$' instead of '$\mathit{stop}(a,.)(t)$'. Remark that a term always contains one variable. We use $x, y, z, \ldots$ to denote the variables.

Let $\mathcal{I}^0$ be an interpretation of $\mathcal{L}_A$ with domain S. We make the following refinements concerning the (given) functions *action* and *effect*: For each $a \in A_{\delta\tau}$ let the function $\mathit{effect}(.,a) : S \rightarrow S$ be determined by the function *effect* and let a predicate $\mathit{stop}(a,.) \subseteq S$ be defined such that $\mathit{stop}(a,s)$ holds if and only if $\mathit{action}(a,s) = \delta$. We consider the functions $\mathit{effect}(.,a)$ and the unary predicates $\mathit{stop}(a,.)$ as interpretations of the equally named $\mathcal{L}_A$-expressions $\mathit{effect}(.,a)$ respectively $\mathit{stop}(a,.)$. Logical formulas are interpreted as usual. As an example, assume that $\mathit{effect}(s,a) = s$ for all $s \in S$ and some $a \in A$. Clearly $\mathcal{I}^0 \models \mathit{stop}(b,x) \leftrightarrow \mathit{stop}(b, \mathit{effect}(x,a))$ for all $b \in A_{\delta\tau}$. Note that $\mathcal{L}_A$-variables refer to elements of S, in contradistinction to the process variables $x, y, \ldots$ introduced in section 1.1. However, this will not cause any confusion.

For each subset of S we add a unary predicate symbol to $\mathcal{L}_A$ which we interpret as the predicate over S satisfied exactly by this subset. Let $\alpha, \beta, \ldots$ be syntactic variables for these added symbols. We will denote the predicate over S referred to by a syntactic variable α with an equally named symbol α. We call this expansion of $\mathcal{L}_A$ the *language of assertions*, $\mathcal{L}_{A,S}$, and refer to the interpretation of $\mathcal{L}_{A,S}$ by the symbol $\mathcal{I}$. The set of logical formulas true in $\mathcal{I}$ is denoted as $Tr_{\mathcal{I}}$ and we write $\models^{\mathcal{I}} \varphi$ if $\varphi \in Tr_{\mathcal{I}}$. Let $\varphi, \psi, \ldots$ denote the logical formulas of $\mathcal{L}_{A,S}$.

Now a syntactical definition of a partial correctness assertion can be given:

Definition 2.3.1 *A* partial correctness assertion *is an expression of the form*

$$\{\varphi(x)\}\, p\, \{\psi(x)\}$$

where p is a closed process expression, and $\varphi(x), \psi(x)$ are $\mathcal{L}_{A,S}$–formulas containing one variable x.

Partial correctness assertions are not subject to Boolean operations, so we may omit the variable concerned. Note that for any logical formula $\varphi(x)$ containing one variable x, there is a predicate symbol α_φ such that $\models^I \varphi(x) \leftrightarrow \alpha_\varphi(x)$ by definition of $\mathcal{L}_{A,S}$. Therefore we will only consider partial correctness assertions of the form $\{\alpha\}\ p\ \{\beta\}$. Partial correctness assertions are interpreted in the following way:

Definition 2.3.2 *A partial correctness assertion $\{\alpha\}\, p\, \{\beta\}$ is true in S,*

$$S \models \{\alpha\}\, p\, \{\beta\}$$

if for all $s \in S$, $\sigma \in (A_\tau)^*$ we have

$$\alpha(s) \textit{ and } (p, s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')) \quad \Longrightarrow \quad \beta(s').$$

So the truth of a partial correctness assertion $\{\alpha\}\ p\ \{\beta\}$ in a structure S expresses the fact that any successful execution of p in an initial state satisfying α, results in a final state satisfying β. A semantical relation based on partial correctness assertions can be defined as follows:

Definition 2.3.3 *We call two closed process expressions p and q* equivalent under partial correctness in *S,*

$$S \models p =_{pc} q$$

if for all predicates α, β over S we have

$$S \models \{\alpha\}\, p\, \{\beta\} \quad \Longleftrightarrow \quad S \models \{\alpha\}\, q\, \{\beta\}.$$

Obviously $=_{pc}$ is an equivalence relation in all structures S, we show that it is also a congruence:

Theorem 2.3.4 *For all structures S the relation $=_{pc}$ is a congruence with respect to the operators involved.*

Proof. Fix S. Observe that if $(p, s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')$, then there is a $\sigma' \in (A_\tau)^*$ such that $(p, s) \overset{\sigma'}{\twoheadrightarrow} (\surd, s')$ without any application of one of the τ-laws in the composing transitions. Call the latter sort of reductions 'operational'.

We prove the theorem by inspection: Fix S and suppose $S \models p =_{pc} p'$, $S \models q =_{pc} q'$. We have to show $S \models p \Box q =_{pc} p' \Box q'$ for $\Box \in \{+, \cdot\}$ and $\pi_n(p) =_{pc} \pi_n(p')$. As an example we consider alternative composition: It is sufficient to show that if we have an operational reduction $(p + q, s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')$, then there is an operational reduction $(p' + q', s) \overset{\rho}{\twoheadrightarrow} (\surd, s')$. This follows easy: Suppose the first transition in our reduction, say $(p + q, s) \xrightarrow{a} (r, s'')$, is a consequence of $(p, s) \xrightarrow{a} (r, s'')$. By the induction hypothesis we have $(p', s) \overset{\rho}{\twoheadrightarrow} (\surd, s')$ for some string $\rho \in (A_\tau)^*$, so using the first transition of this (operational) reduction we derive $(p' + q', s) \overset{\rho}{\twoheadrightarrow} (\surd, s')$. □

2.4 Proof systems

A proof system H is a finite set of (schemes of) axioms and rules which we can use to derive partial correctness assertions. We write $\Gamma \vdash^H \{\alpha\} p \{\beta\}$ with Γ a set of logical formulas, if there is a derivation of $\{\alpha\}\, p\, \{\beta\}$ in a proof system H in which the elements of Γ may be used. The introduction of a proof system H will always be accompanied with a specification of a *process domain* $\mathcal{P}_H$, i.e. a set of closed process expressions of which partial correctness assertions are to be constructed. In the context of a fixed process domain $\mathcal{P}_H \subseteq \mathcal{P}$, we use $p, q, \ldots$ as syntactic variables over this domain. If H is fixed, we omit the superscript H in $\vdash^H$.

2.4.1 Soundness and completeness

We call a proof system H *sound* if for all structures $\mathcal{S}$ and all partial correctness assertions $\{\alpha\}\,p\,\{\beta\}$ over $\mathcal{P}_H$ we have

$$Tr_{\mathcal{S}} \vdash^H \{\alpha\}\,p\,\{\beta\} \implies \mathcal{S} \models \{\alpha\}\,p\,\{\beta\}$$

i.e. every partial correctness assertion derivable in H (using $Tr_{\mathcal{S}}$) is true in $\mathcal{S}$.

The proof system H is *(relatively) complete* if the converse holds as well for all structures $\mathcal{S}$:

$$\mathcal{S} \models \{\alpha\}\,p\,\{\beta\} \iff Tr_{\mathcal{S}} \vdash^H \{\alpha\}\,p\,\{\beta\}$$

that is, a partial correctness assertion $\{\alpha\}\,p\,\{\beta\}$ is true in $\mathcal{S}$ if and only if $\{\alpha\}\,p\,\{\beta\}$ is derivable in H (using $Tr_{\mathcal{S}}$).

3 A semantical tour

In this section we compare our semantical notions with other process algebra semantics. We show that for all structures $\mathcal{S}$ the relation $=_{se}$ provides us with a 'proper' semantics, related to a well-known semantics concerning closed process expressions which are *not* state labelled. As a consequence we conclude that our semantical notions satisfy the axioms of table 2, a fact that will be used to prove our main result. We will also compare the semantics of the relation $=_{pc}$ with 'complete trace semantics'.

3.1 Action rules

A statement $\mathcal{S} \models p =_{se} q$ refers to a well-known semantical area: 'Bisimulation semantics of transition systems', where transition systems are considered to be generated by a related set of action rules. In table 5 we present these rules, which will be referred to simply as 'action rules'. Again the reflexive and transitive closure of the transition relations $\xrightarrow{a}$ and $\xrightarrow{a} \surd$ are denoted by $\xrightarrow{\sigma}\!\!\!\!\rightarrow$, respectively $\xrightarrow{\sigma}\!\!\!\!\rightarrow \surd$. Note that the action rules are related to the effect rules in the following sense: If the state components in the effect rules are deleted, we find action rules. We write $[\![p]\!]$ for the transition system (generated by these related action rules in the way as described in section 2.2) associated to p, and $[\![p]\!] \leftrightarrows [\![q]\!]$ if $[\![p]\!]$ and $[\![q]\!]$ are bisimilar[3]. In order to relate the reflexive and transitive closures of action rules and effect rules we generalize for each fixed structure $\mathcal{S}$ the functions *action* and *effect*:

- *action* : $(A_{\delta\tau})^* \times S \rightarrow (A_{\delta\tau})^*$ such that $\sigma(s) \overset{\text{def}}{=} \begin{cases} a(s) \star \sigma'(s(a)) & \text{if } a \in A_{\delta\tau} \text{ and } \sigma \equiv a \star \sigma' \\ \lambda & \text{if } \sigma \equiv \lambda \end{cases}$

- *effect*: $S \times (A_{\delta\tau})^* \rightarrow S$ such that $s(\sigma) \overset{\text{def}}{=} \begin{cases} s(a)(\sigma') & \text{if } a \in A_{\delta\tau} \text{ and } \sigma \equiv a \star \sigma' \\ s & \text{if } \sigma \equiv \lambda \end{cases}$

It can be easily proved (by induction on the length of strings) that for all $s \in S$, $\sigma_1, \sigma_2 \in (A_\tau)^*$: $s(\sigma_1\sigma_2) = s(\sigma_1)(\sigma_2)$, $\sigma_1\sigma_2(s) \equiv \sigma_1(s) \star \sigma_2(s(\sigma_1))$. Note that for all structures $\mathcal{S}$ and $\sigma \in (A_{\delta\tau})^*$ we have $s(\sigma) = s(\widetilde{\sigma})$ because τ is inert.

Finally we present two useful relations between effect reductions and action reductions. Let $\surd q$ denote either some element of $\mathcal{P}$ or the symbol $\surd$, then we have by definition of these generalized functions the following relations:

- If $(p, s) \xrightarrow{\sigma}\!\!\!\!\rightarrow (\surd q, s')$ for some $\sigma \in (A_\tau)^*$, then for some $\rho \in (A_\tau)^*$ we have $p \xrightarrow{\rho}\!\!\!\!\rightarrow \surd q$ and $\widetilde{\rho(s)} \equiv \widetilde{\sigma}$, $s(\rho) = s'$.

- If $p \xrightarrow{\sigma}\!\!\!\!\rightarrow \surd q$ for some $\sigma \in (A_\tau)^*$, then $(p, s) \xrightarrow{\sigma(s)}\!\!\!\!\rightarrow (\surd q, s(\sigma))$ if $\sigma(s)$ is δ-free.

[3] See eg. [6] for these semantics.

$a \in A_\tau$:	$a \xrightarrow{a} \surd$		
$+$:	$\dfrac{x \xrightarrow{a} x'}{x+y \xrightarrow{a} x'}$	$\dfrac{x \xrightarrow{a} \surd}{x+y \xrightarrow{a} \surd}$	
	$\dfrac{y \xrightarrow{a} y'}{x+y \xrightarrow{a} y'}$	$\dfrac{y \xrightarrow{a} \surd}{x+y \xrightarrow{a} \surd}$	
$\cdot$:	$\dfrac{x \xrightarrow{a} x'}{xy \xrightarrow{a} x'y}$	$\dfrac{x \xrightarrow{a} \surd}{xy \xrightarrow{a} y}$	
π_n :	$\dfrac{x \xrightarrow{a} y}{\pi_{n+1}(x) \xrightarrow{a} \pi_n(y)}$	$\dfrac{x \xrightarrow{a} \surd}{\pi_{n+1}(x) \xrightarrow{a} \surd}$	(if $a \neq \tau$)
	$\dfrac{x \xrightarrow{\tau} y}{\pi_n(x) \xrightarrow{\tau} \pi_n(y)}$	$\dfrac{x \xrightarrow{\tau} \surd}{\pi_n(x) \xrightarrow{\tau} \surd}$	
recursion :	$\dfrac{<t_x \mid E> \xrightarrow{a} y}{<x \mid E> \xrightarrow{a} y}$	$\dfrac{<t_x \mid E> \xrightarrow{a} \surd}{<x \mid E> \xrightarrow{a} \surd}$	
$\tau-$laws :	$a \xrightarrow{a} \tau$		
	$\dfrac{x \xrightarrow{\tau} y \quad y \xrightarrow{a} z}{x \xrightarrow{a} z}$	$\dfrac{x \xrightarrow{\tau} y \quad y \xrightarrow{a} \surd}{x \xrightarrow{a} \surd}$	
	$\dfrac{x \xrightarrow{a} y \quad y \xrightarrow{\tau} z}{x \xrightarrow{a} z}$	$\dfrac{x \xrightarrow{a} y \quad y \xrightarrow{\tau} \surd}{x \xrightarrow{a} \surd}$	

Table 5: Action rules

3.2 On semantical equivalence

We show that for all structures $\mathcal{S}$ the relation $=_{se}$ defines a semantics which can be related to a more usual semantics of closed process expressions, namely that of transition systems generated by the action rules of table 5.

Theorem 3.2.1 *If two closed process expressions p and q are identified in 'bisimulation semantics of transition systems', then $\mathcal{S} \models p =_{se} q$ for all structures $\mathcal{S}$.*

Proof. Suppose $[\![p]\!] \underline{\leftrightarrow} [\![q]\!]$. Fix some $\mathcal{S}$ and $s \in \mathcal{S}$. Now the system $ts((p,s))$ can be constructed from the system $[\![p]\!]$:

1. Label the root of the construction with (p,s).
2. If (p',s') is a node in the construction, $p' \xrightarrow{a} p''$ is a transition in $[\![p]\!]$ and $a(s') \neq \delta$, then add the transition $(p',s') \xrightarrow{a(s')} (p'', s'(a))$ to the construction.
3. If (p',s') is a node in the construction, $p' \xrightarrow{a} \surd$ is a transition in $[\![p]\!]$ and $a(s') \neq \delta$, then add the transition $(p',s') \xrightarrow{a(s')} (\surd, s'(a))$ to the construction.
4. Add all transitions which can be freshly generated by the τ-laws to the construction ($a(s')$ may be τ if $a \in A$).

This construction yields the system $ts((p,s))$, and since it is based exclusively on the transitions in $[\![p]\!]$, the bisimilarity of $[\![p]\!]$ and $[\![q]\!]$ implies that $ts((p,s)) \underline{\leftrightarrow} ts((q,s))$, and thus $\mathcal{S} \models p =_{se} q$. □

As an immediate result we have the following

Corollary 3.2.2 *Write $\mathcal{S} \models^{se} p = q$ for $\mathcal{S} \models p =_{se} q$, then for all structures $\mathcal{S}$ we have $\mathcal{S} \models^{se}$* $\mathrm{BPA}_{\delta\tau} + \mathrm{REC} + \mathrm{PR} + \mathrm{B} + \mathrm{AIP}^-$ (see [6]).

3.3 On equivalence under partial correctness

To begin with, we present a useful result concerning the relations $=_{se}$ and $=_{pc}$. Furthermore we compare the relations $=_{pc}$ and $=_{ctr}$.

Theorem 3.3.1 *If for two closed process expressions p and q we have that $\mathcal{S} \models p =_{se} q$, then $\mathcal{S} \models p =_{pc} q$.*

Proof. Fix $\mathcal{S}$. Suppose $\mathcal{S} \models \{\alpha\}\, p\, \{\beta\}, \alpha(s)$ and $(q,s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')$, then there is a string $\rho \in (A_\tau)^*$ such that $q \overset{\rho}{\twoheadrightarrow} \surd$ and $s(\rho) = s'$, $\widetilde{\rho(s)} \equiv \tilde{\sigma}$. Obviously $\widetilde{\rho(s)}$ is δ-free, so $\rho(s)$ is as well. By $[\![p]\!] \underline{\leftrightarrow} [\![q]\!]$ we have $p \overset{\rho}{\twoheadrightarrow} \surd$, so $(p,s) \overset{\rho(s)}{\twoheadrightarrow} (\surd, s(\rho))$ and therefore $\beta(s')$ holds, so $\mathcal{S} \models \{\alpha\}\, q\, \{\beta\}$. □

Remark 3.3.2 *The converse of theorem 3.3.1 does not hold: $\mathcal{S} \models a\delta =_{pc} \delta$ for neither can terminate, but $\mathcal{S} \models a\delta =_{se} \delta$ $(a \in A_\tau)$ only if $\forall s \in \mathcal{S}(a(s) = \delta)$.*

For each $p \in \mathcal{P}$ we now consider the set of complete traces of p (see section 1.2 for the definition of a complete trace.) We present two lemmas and prove a final result which relates the semantics of $=_{pc}$ to 'complete trace semantics'.

Lemma 3.3.3 *If $\sigma \in tr_c(p)$, then there is a string $\rho \in (A_\tau)^*$ such that $\tilde{\rho} \equiv \sigma$ and $p \overset{\rho}{\twoheadrightarrow} \surd$.*

Proof. By definition of tr_c. For the recursive case observe that by definition of the action rules we have $\pi_n(p) \overset{\rho}{\twoheadrightarrow} \surd$ implies $p \overset{\rho}{\twoheadrightarrow} \surd$. □

Lemma 3.3.4 *If $p \overset{\sigma}{\twoheadrightarrow} \surd$, then $\widetilde{\sigma} \in tr_c(p)$.*

Proof. This can be proved in four steps ($a \in A_\tau$):

1. $p \xrightarrow{a} q \quad\Longrightarrow\quad tr_c(aq) \subseteq tr_c(p)$ (By induction on the length of the proof of $p \xrightarrow{a} q$.)

2. $p \xrightarrow{a} \surd \quad\Longrightarrow\quad \widetilde{a} \in tr_c(p)$ (Again by induction and step 1.)

3. $p \overset{\sigma}{\twoheadrightarrow} q \quad\Longrightarrow\quad \{\widetilde{\sigma}\rho \mid \rho \in tr_c(q)\} \subseteq tr_c(p)$ (By definition of $\overset{\sigma}{\twoheadrightarrow}$ and steps 1 and 2.)

4. $p \overset{\sigma}{\twoheadrightarrow} \surd \quad\Longrightarrow\quad \widetilde{\sigma} \in tr_c(q)$ (By definition of $\overset{\sigma}{\twoheadrightarrow}$ and step 3.) □

As an immediate result we have the following

Theorem 3.3.5 *If for two closed process expressions p and q we have that $p =_{ctr} q$, then $S \models p =_{pc} q$ for all S.*

Proof. Fix S and suppose $S \models \{\alpha\}\, p\, \{\beta\}$, $\alpha(s)$ and $(q, s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')$. There is a string $\rho \in (A_\tau)^*$ such that $q \overset{\rho}{\twoheadrightarrow} \surd$ and $s(\rho) = s'$, $\widetilde{\rho(s)} \equiv \widetilde{\sigma}$ (so $\rho(s)$ is δ-free). By lemma 3.3.4 we have $\widetilde{\rho} \in tr_c(q)$ and thus $\widetilde{\rho} \in tr_c(p)$. So by lemma 3.3.3 there is a string $\nu \in (A_\tau)^*$ such that $\widetilde{\nu} \equiv \widetilde{\rho}$ (so $\nu(s)$ is δ-free) and $p \overset{\nu}{\twoheadrightarrow} \surd$. Therefore $(p, s) \overset{\nu(s)}{\twoheadrightarrow} (\surd, s(\nu))$ and $s(\nu) = s(\rho) = s'$. As a consequence $\beta(s')$ holds, and thus $S \models \{\alpha\}\, q\, \{\beta\}$, as was to be proved. □

Corollary 3.3.6 *If $p =_{ctr} q$ or $S \models p =_{se} q$, and $S \models \{\alpha\}\, p\, \{\beta\}$, then $S \models \{\alpha\}\, q\, \{\beta\}$. More generally: $S \models^{pc}$* BPA$_{\delta\tau}$ + REC + PR + B + AIP$^-$ + CTR *for all structures S (see corollary 3.2.2 and theorems 3.3.1, 3.3.5).*

4 Finite processes and Hoare's logic

In this section partial correctness assertions over finite processes are studied. A Hoare-like proof system F for deriving such partial correctness assertions is defined and will be proved sound and complete.

4.1 The proof system F

Let the process domain $\mathcal{P}_F$ be specified inductively as the least set satisfying:

- $a \in \mathcal{P}_F$ if $a \in A_{\delta\tau}$

- $p + q \in \mathcal{P}_F$ if $p, q \in \mathcal{P}_F$

- $pq \in \mathcal{P}_F$ if $p, q \in \mathcal{P}_F$

Note that we do not allow expressions containing occurrences of a π_n-operator. The proof system F is defined as follows:

I *axioms* $(a \in A_{\delta\tau})$
$$\frac{\left(\neg stop(a,x) \wedge \alpha(x)\right) \;\rightarrow\; \beta(effect(x,a))}{\{\alpha\}\, a\, \{\beta\}}$$

II *alternative composition*
$$\frac{\{\alpha\}\, p\, \{\beta\} \quad \{\alpha\}\, q\, \{\beta\}}{\{\alpha\}\, p+q\, \{\beta\}}$$

III *sequential composition*
$$\frac{\{\alpha\}\, p\, \{\beta\} \quad \{\beta\}\, q\, \{\gamma\}}{\{\alpha\}\, pq\, \{\beta\}}$$

IV *consequence*
$$\frac{\alpha \rightarrow \alpha' \quad \{\alpha'\}\, p\, \{\beta'\} \quad \beta' \rightarrow \beta}{\{\alpha\}\, p\, \{\beta\}}$$

4.2 F is complete

Lemma 4.2.1 *The proof system F is sound.*

Proof. Induction on the length of derivations. The soundness of the rules I and IV follows easy. The soundness of rule II follows by a simple induction on the length of derivations. Referring to result 1 in section 2.1 we conclude that rule III is sound. □

Theorem 4.2.2 *The proof system F is complete.*

Proof. By lemma 4.2.1 we only have to show that $S \models \{\alpha\}\, p\, \{\beta\} \implies Tr_S \vdash \{\alpha\}\, p\, \{\beta\}$. This can be proved by induction on the structure of process expressions. Suppose $S \models \{\alpha\}\, p\, \{\beta\}$.

$p \equiv a \in A_{\delta\tau}$: Now Tr_S contains the formula

$$\left(\neg stop(a,x) \wedge \alpha(x)\right) \;\rightarrow\; \beta(effect(x,a))$$

for, if $\neg stop(a,s) \wedge \alpha(s)$, then $\beta(s(a))$ by supposition, and else the formula holds trivially in S. By the axiom I we derive $Tr_S \vdash \{\alpha\}\, a\, \{\beta\}$.

$p \equiv qr$: Let γ be the predicate over S such that

$$\gamma(s') \iff \alpha(s) \wedge ((q,s) \longrightarrow\!\!\!\!\!\rightarrow (\surd, s'))$$

Referring to result 2 in section 2.1 we state that $S \models \{\alpha\}\, q\, \{\gamma\}$, $S \models \{\gamma\}\, r\, \{\beta\}$ and by the induction hypothesis and rule III we derive $Tr_S \vdash \{\alpha\}\, qr\, \{\beta\}$.

$p \equiv q+r$: Note that $S \models \{\alpha\}\, q\, \{\beta\}$, $S \models \{\alpha\}\, r\, \{\beta\}$. By the induction hypothesis and rule II we derive $Tr_S \vdash \{\alpha\}\, q+r\, \{\beta\}$. □

4.3 Derived proof rules

By means of the completeness of F we can easily derive new proof rules:

substitution $$\frac{\{\alpha\}\,p\,\{\beta\} \qquad \mathrm{BPA}_{\delta\tau} \vdash p = q}{\{\alpha\}\,q\,\{\beta\}}$$

Proof. Let S be fixed. Because $S \models p =_{pc} q$ (see corollary 3.3.6) we have that $S \models \{\alpha\}\,q\,\{\beta\}$. By the completeness of F we conclude $Tr_S \vdash \{\alpha\}\,q\,\{\beta\}$. □

disjunction $$\frac{\{\alpha\}\,p\,\{\gamma\} \quad \{\beta\}\,p\,\{\gamma\}}{\{\alpha \vee \beta\}\,p\,\{\gamma\}}$$

Proof. If for a structure S we have that $S \models \{\alpha\}\,p\,\{\gamma\}$, $S \models \{\beta\}\,p\,\{\gamma\}$, then we easily conclude $S \models \{\alpha \vee \beta\}\,p\,\{\gamma\}$ and by completeness of F this implies $Tr_S \vdash \{\alpha \vee \beta\}\,p\,\{\gamma\}$. □

conjunction $$\frac{\{\alpha\}\,p\,\{\beta\} \quad \{\alpha\}\,p\,\{\gamma\}}{\{\alpha\}\,p\,\{\beta \wedge \gamma\}}$$

Proof. Likewise.

5 Semi linear systems and Hoare's logic

In this section we introduce a proof system G dealing with a restricted format of guarded systems, which will be proved sound and complete (with the use of PR). Both these proofs are considerably more complicated than in the case of the proof system F.

5.1 The proof system G

Firstly we introduce a restricted format of guarded systems:

Definition 5.1.1 *A guarded system $E = \{x = t_x \,|\, x \in V_E\}$ will be called* semi linear *if for all $x \in V_E$ the expressions t_x contain only summands of either the form py ($y \in V_E$), or the form q, where p is a closed process expression of depth 1, containing at most the sequential operator, and q a closed process expression of depth ≤ 1, containing at most the sequential operator.*

Eg. $E = \{X = \tau b \tau^3 X + a\delta X + \tau\}$ is a semi linear system if $a, b \in A$. We specify the process domain $\mathcal{P}_G$ as follows:

- $a \in \mathcal{P}_G$ if $a \in A_{\delta\tau}$
- $<x\,|\,E> \in \mathcal{P}_G$ if $E = \{x = t_x \,|\, x \in V_E\}$ is a semi linear system
- $p + q \in \mathcal{P}_G$ if $p, q \in \mathcal{P}_G$
- $pq \in \mathcal{P}_G$ if $p, q \in \mathcal{P}_G$

We now want to involve recursively specified process expressions in partial correctness assertions. In order to construct a proof rule introducing partial correctness assertions of the form

$$\{\alpha\} <x\,|\,E> \{\beta\}$$

we take a more 'abstract' point of view:

Definition 5.1.2 *An* abstract partial correctness assertion *over some process domain $\mathcal{P}_X$ is an expression $\{\alpha\}\, t\, \{\beta\}$, where t is an expression over the syntax of $\mathcal{P}_X$ which may contain variables.*

Though we will not define how to interpret abstract partial correctness assertions, we may use them as 'axioms' in derivations. We extend the proof system F to a proof system G defined over $\mathcal{P}_G$ by adding the following 'recursion rule', where $E = \{x = t_x \,|\, x \in V_E\}$ is a semi linear system:

$$\text{V} \quad \textit{recursion} \quad \frac{\big\{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\big\} \vdash \big\{\{\alpha_x\}\, t_x\, \{\beta_x\} \mid x \in V_E\big\}}{\big\{\{\alpha_x\} <x|E> \{\beta_x\} \mid x \in V_E\big\}}$$

We adopt the convention that if for sets Υ and Θ and some inference system referred to by '$\vdash$' we have that $\Upsilon \vdash \theta$ for all $\theta \in \Theta$, then we write $\Upsilon \vdash \Theta$. In particular $\Upsilon \vdash \{\theta\} \iff \Upsilon \vdash \theta$.

5.2 G is complete

We cannot prove that G is sound by showing that its axioms are valid and its rules are sound because of the nature of the recursion rule. As a solution to this problem we follow the strategy presented in [1]: We define a related proof system K which will be proved sound in the usual way. By showing that the soundness of K implies the soundness of G we are done. The proof system K manipulates *correctness phrases* of the form $\Phi \longrightarrow \Psi$, where the symbols Φ and Ψ denote (possibly empty) sets of partial correctness assertions, which are either abstract, or not.

We define K on $\mathcal{P}_G$ in table 6, where $E = \{x = t_x \mid x \in V_E\}$ denotes a semi linear system. Correctness phrases are interpreted as follows:

Definition 5.2.1 *A correctness phrase $\Phi \longrightarrow \Psi$ is true in a structure $\mathcal{S}$,*

$$\mathcal{S} \models \Phi \longrightarrow \Psi$$

if for any substitution θ taking abstract partial correctness assertions to partial correctness assertions over $\mathcal{P}$ (sic!), we have that $\mathcal{S} \models \theta(\Phi)$ implies that $\mathcal{S} \models \theta(\Psi)$.

Lemma 5.2.2 *The proof system K is sound.*

Proof. Induction on the length of derivations. We check the soundness of rule V* (the other cases are straightforward).

Let $\mathcal{S}$ be fixed and $E = \{x = t_x \,|\, x \in V_E\}$ a semi linear system. Suppose

$$\mathcal{S} \models \big\{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\big\} \longrightarrow \big\{\{\alpha_x\}\, t_x\, \{\beta_x\} \mid x \in V_E\big\}$$

For a start we prove that

$$\forall n \in \mathbb{N}\big(\mathcal{S} \models \big\{\{\alpha_x\}\, \pi_n(<x|E>)\, \{\beta_x\} \mid x \in V_E\big\}\big).$$

$n = 0$: Assume that for some $x_0 \in V_E$ we have $\alpha_{x_0}(s)$ and

$$(\pi_0(<x_0\,|\,E>), s) \xrightarrow{\sigma} (\surd, s').$$

This implies that $(\pi_0(<t_{x_0}\,|\,E>), s) \xrightarrow{\sigma} (\surd, s')$. By the guardedness of E we conclude that there is a summand q of t_{x_0} not containing variables such that $(q, s) \xrightarrow{\sigma} (\surd, s')$. Now we have by supposition that $\mathcal{S} \models \{\alpha_{x_0}\}\, t_{x_0}(\,\overline{\delta}\,)\, \{\beta_{x_0}\}$ (substitute δ for all $x \in V_E$). In particular $\mathcal{S} \models \{\alpha_{x_0}\}\, q\, \{\beta_{x_0}\}$ and therefore $\beta_{x_0}(s')$ holds. We conclude

$$\mathcal{S} \models \big\{\{\alpha_x\}\, \pi_0(<x|E>)\, \{\beta_x\} \mid x \in V_E\big\}.$$

I*	*axioms* $(a \in A_{\delta\tau})$	$\dfrac{\left(\neg stop(a,x) \wedge \alpha(x)\right) \;\rightarrow\; \beta(effect(x,a))}{\Phi \longrightarrow \left\{\{\alpha\}\, a\, \{\beta\}\right\}}$
II*	*alternative composition*	$\dfrac{\Phi \longrightarrow \left\{\{\alpha\}\, p\, \{\beta\}, \{\alpha\}\, q\, \{\beta\}\right\}}{\Phi \longrightarrow \left\{\{\alpha\}\, p+q\, \{\beta\}\right\}}$
III*	*sequential composition*	$\dfrac{\Phi \longrightarrow \left\{\{\alpha\}\, p\, \{\beta\}, \{\beta\}\, q\, \{\gamma\}\right\}}{\Phi \longrightarrow \left\{\{\alpha\}\, pq\, \{\beta\}\right\}}$
IV*	*consequence*	$\dfrac{\alpha \rightarrow \alpha' \quad \Phi \longrightarrow \left\{\{\alpha'\}\, p\, \{\beta'\}\right\} \quad \beta' \rightarrow \beta}{\Phi \longrightarrow \left\{\{\alpha\}\, p\, \{\beta\}\right\}}$
V*	*recursion*	$\dfrac{\left\{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\right\} \longrightarrow \left\{\{\alpha_x\}\, t_x\, \{\beta_x\} \mid x \in V_E\right\}}{\Phi \longrightarrow \left\{\{\alpha_x\} <x \mid E> \{\beta_x\} \mid x \in V_E\right\}}$
VI*	*collection*	$\dfrac{\Phi \longrightarrow \Psi \quad \Phi \longrightarrow \Psi'}{\Phi \longrightarrow \Psi \cup \Psi'}$

Table 6: The proof system K

$n+1$: Assume that for some $x_0 \in V_E$ we have $\alpha_{x_0}(s)$ and

$$(\pi_{n+1}(<x_0\,|\,E>), s) \xrightarrow{\sigma} (\surd, s').$$

Let $t_{x_0} \equiv \sum p_i y_i \ + \sum q_j$, where the y_i are in V_E. Now we have that

$$(\pi_{n+1}(q_{j0}), s) \xrightarrow{\sigma} (\surd, s')$$

for a summand q_{j0} of t_{x_0} because $(q_{j_0}, s) \xrightarrow{\sigma} (\surd, s')$, or

$$(\pi_{n+1}(<p_{i_0}y_{i_0}\,|\,E>), s) \xrightarrow{\sigma} (\surd, s')$$

for a summand $p_{i_0}y_{i_0}$ of t_{x_0}, so there must be an atomic action $a \in A$ such that

$$(\pi_{n+1}(<p_{i_0}y_{i0}\,|\,E>), s) \xrightarrow{a} (\pi_n(<y_{i_0}\,|\,E>), s'') \xrightarrow{\sigma'} (\surd, s')$$

for some $s'' \in S$ and $a\sigma' \equiv \sigma$ because E is a semi linear system. In both cases we may conclude that

$$(t_{x_0}(\,\overline{\pi_n(<x\,|\,E>)}\,), s) \xrightarrow{\sigma} (\surd, s').$$

By the induction hypothesis we have $S \models \{\{\alpha_x\}\, \pi_n(<x|\,E>)\, \{\beta_x\} \mid x \in V_E\}$, so by supposition we have in particular that

$$S \models \{\alpha_{x_0}\}\, t_{x_0}(\,\overline{\pi_n(<x\,|\,E>)}\,)\, \{\beta_{x_0}\}$$

and thus $\beta_{x_0}(s')$ holds. We conclude

$$S \models \{\{\alpha_x\}\, \pi_{n+1}(<x|\,E>)\, \{\beta_x\} \mid x \in V_E\}.$$

Next we have to show that if $\forall n \in \mathbb{N}(S \models \{\alpha\}\, \pi_n(p)\, \{\beta\})$, then $S \models \{\alpha\}\, p\, \{\beta\}$. Suppose $\alpha(s)$ and $(p, s) \xrightarrow{\sigma} (\surd, s')$. By definition of the effect rules it follows easily that for n sufficiently large $(\pi_n(p), s) \xrightarrow{\sigma} (\surd, s')$, so $\beta(s')$ holds, which shows that $S \models \{\alpha\}\, p\, \{\beta\}$.

Now we may conclude $S \models \{\{\alpha_x\} <x|\,E> \{\beta_x\} \mid x \in V_E\}$, and therefore

$$S \models \Phi \longrightarrow \{\{\alpha_x\} <x|\,E> \{\beta_x\} \mid x \in V_E\}$$

which completes our proof. □

In order to prove the soundness of the proof system G we first show that the soundness of K implies the soundness of G.

Lemma 5.2.3 *If* $\Gamma, \Phi \vdash^G \Psi$, *then* $\Gamma \vdash^K \Phi \longrightarrow \Psi$.

Proof. Induction on the length of derivations. As an example we check the application of rule V (the other cases are again straightforward): Suppose $E = \{x = t_x \,|\, x \in V_E\}$ is a semi linear system and

$$\Gamma, \Phi \vdash^G \{\{\alpha_x\} <x|\,E> \{\beta_x\} \mid x \in V_E\}$$

as a result of V, thus

$$\Gamma, \{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\} \ \vdash^G \{\{\alpha_x\}\, t_x\, \{\beta_x\} \mid x \in V_E\}.$$

By the induction hypothesis we have $\Gamma \vdash^K \{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\} \longrightarrow \{\{\alpha_{x_0}\}\, t_{x_0}\, \{\beta_{x_0}\}\}$ for all $x_0 \in V_E$, so by VI* we derive

$$\Gamma \vdash^K \{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\} \longrightarrow \{\{\alpha_x\}\, t_x\, \{\beta_x\} \mid x \in V_E\}$$

By rule V* we derive $\Gamma \vdash^K \Phi \longrightarrow \{\{\alpha_x\} <x|\,E> \{\beta_x\} \mid x \in V_E\}$. □

Lemma 5.2.4 *The proof system G is sound.*

Proof. Fix some structure S and suppose $Tr_S \vdash^G \{\alpha\}\, p\, \{\beta\}$, then by lemma 5.2.3 we have $Tr_S \vdash^K \emptyset \longrightarrow \big\{\{\alpha\}\, p\, \{\beta\}\big\}$. By the soundness of K we conclude $S \models \{\alpha\}\, p\, \{\beta\}$, as was to be proved. □

Before proving the completeness of G, we take a closer look at a statement $S \models \{\alpha\} {<}x \,|\, E{>} \{\beta\}$. In the following lemma we show that such a statement implies $Tr_S \vdash \{\alpha\} {<}x\,|\,E{>} \{\beta\}$.

Lemma 5.2.5 *If for some S and semi linear system $E = \{x = t_x \,|\, x \in V_E\}$ we have for some $x_0 \in V_E$ that*

$$S \models \{\alpha\} {<}x_0 \,|\, E{>} \{\beta\}$$

then also

$$Tr_S \vdash \{\alpha\} {<}x_0 \,|\, E{>} \{\beta\}.$$

Proof. We construct predicates α_x $(x \in V_E)$ as follows:

$$\alpha_x(s) \iff \text{There are } s' \in S, \sigma \in (A_\tau)^* \text{ such that} ({<}x_0\,|\,E{>}, s') \overset{\sigma}{\twoheadrightarrow} ({<}x\,|\,E{>}, s) \text{ and } \alpha(s').$$

Observe that

- $\alpha \rightarrow \alpha_{x_0} \in Tr_S$
- $S \models \{\alpha_{x_0}\} {<}x_0\,|\,E{>} \{\beta\}$

We prove that

$$Tr_S, \big\{\{\alpha_x\}\, x\, \{\beta\} \,|\, x \in V_E\big\} \vdash \big\{\{\alpha_x\}\, t_x\, \{\beta\} \,|\, x \in V_E\big\}.$$

Define Θ as $Tr_S \cup \big\{\{\alpha_x\}\, x\, \{\beta\} \,|\, x \in V_E\big\}$. Fix $x_1 \in V_E$. It is sufficient to show that

1. For any summand px of t_{x_1} $(x \in V_E)$ we have $\Theta \vdash \{\alpha_{x_1}\}\, px\, \{\beta\}$
2. For any summand q of t_{x_1} we have $\Theta \vdash \{\alpha_{x_1}\}\, q\, \{\beta\}$

ad 1 : We show that $S \models \{\alpha_{x_1}\} p \{\alpha_x\}$ and by completeness of F we conclude $Tr_S \vdash \{\alpha_{x_1}\} p \{\alpha_x\}$ and thus $\Theta \vdash \{\alpha_{x_1}\}\, px\, \{\beta\}$: Define a predicate α_* over S such that:

$$\alpha_*(s) \iff \neg stop(a, s) \wedge \alpha_{x_1}(s) \text{ for all } a \in A_{\delta\tau} \text{ in } p.$$

So α_* selects the elements $s \in S$ satisfying α_{x_1} and $\alpha_*(s) \iff (p, s) \twoheadrightarrow (\surd, .)$ If α_* is empty, we immediately conclude $S \models \{\alpha_*\}\, p\, \{\alpha_x\}$. Suppose α_* is not empty and $\alpha_*(s)$. By construction of α_{x_1} there is an $s' \in S$ such that $\alpha(s')$ and

$$({<}x_0\,|\,E{>}, s') \overset{\sigma}{\twoheadrightarrow} ({<}x_1\,|\,E{>}, s).$$

Because $(p, s) \overset{\rho}{\twoheadrightarrow} (\surd, s'')$ for some string $\rho \in (A_\tau)^*$ and px is a summand of t_{x_1}, we have

$$({<}x_0\,|\,E{>}, s') \overset{\sigma * \rho}{\twoheadrightarrow} ({<}x\,|\,E{>}, s'')$$

and by construction of α_x it follows that $\alpha_x(s'')$, so $S \models \{\alpha_*\}\, p\, \{\alpha_x\}$. We conclude $S \models \{\alpha_{x_1}\}\, p\, \{\alpha_x\}$.

ad 2 : We show that $S \models \{\alpha_{x_1}\}\, q\, \{\beta\}$ and conclude that $\Theta \vdash \{\alpha_{x_1}\}\, q\, \{\beta\}$: Define α_* as above and suppose α_* is not empty. We prove that $S \models \{\alpha_*\} q \{\beta\}$. Assume $\alpha_*(s)$. By construction of α_{x_1} there is an $s' \in S$ such that $\alpha(s')$ and

$$({<}x_0\,|\,E{>}, s') \overset{\sigma}{\twoheadrightarrow} ({<}x_1\,|\,E{>}, s).$$

Because $(q, s) \xrightarrow{\rho}\!\!\!\!\!\rightarrow (\surd, s'')$ for some string $\rho \in (A_\tau)^*$ and q is a summand of t_{x_1}, we have

$$(<\!x_0 \,|\, E\!>, s') \xrightarrow{\sigma * \rho}\!\!\!\!\!\rightarrow (\surd, s'').$$

By the assumption $S \models \{\alpha\} <\!x_0 \,|\, E\!> \{\beta\}$ we have that $\beta(s'')$ holds, so $S \models \{\alpha_*\}\, q\, \{\beta\}$. We conclude $S \models \{\alpha_{x_1}\}\, p\, \{\beta\}$.

We now proved the premisses of the recursion rule, so we conclude that in particular

$$Tr_S \vdash \{\alpha_{x_0}\} <\!x_0 \,|\, E\!> \{\beta\}$$

and because $\alpha \rightarrow \alpha_{x_0} \in Tr_S$ we derive

$$Tr_S \vdash \{\alpha\} <\!x_0 \,|\, E\!> \{\beta\}$$

which completes our proof. Note that if α is the 'empty predicate' the lemma still holds. □

Theorem 5.2.6 *The proof system G is complete.*

Proof. The soundness of G is proved in lemma 5.2.4. In section 4, theorem 4.2.2 we showed that the proof system F was complete by induction on the structure of the process expression p involved in a partial correctness assertion $\{\alpha\}\, p\, \{\beta\}$. As the set $\mathcal{P}_G$ is again inductively specified, we only have to check one more 'basic clause' than in the proof of theorem 4.2.2, namely $p \equiv <\!x \,|\, E\!>$ with $E = \{x = t_x \,|\, x \in V_E\}$ a semi linear system. This has just been done in lemma 5.2.5. □

5.3 Guarded systems and the proof system G

In this section we prove that if we extend $\mathcal{P}_G$ with *all* guarded systems, then the proof system G, with rule V then referring to all guarded systems $E = \{x = t_x \,|\, x \in V_E\}$, is not complete any more. We show this by an example. Crucial is that G is still sound with regard to this extension, as will be proved in section 7.

Example. Consider the structure $S = \langle A, \{s, s'\}, \mathit{action}\ , \mathit{effect}\, \rangle$ with the functions *action* and *effect* defined as follows ($b, c, d \in A$):

- For all $a \in \{b, c, d\}$: $a(s) \overset{\text{def}}{=} a(s') \overset{\text{def}}{=} a$.
- $s(b) \overset{\text{def}}{=} s(c) \overset{\text{def}}{=} s'(d) \overset{\text{def}}{=} s'$, and $s'(b) \overset{\text{def}}{=} s'(c) \overset{\text{def}}{=} s(d) \overset{\text{def}}{=} s$.

We define predicates σ and σ' over $\{s, s'\}$ such that σ is only satisfied by s and σ' only by s'. Consider the guarded system $E = \{X = \tau b X c + d\}$. Now $tr_c(X) = \{b^n d c^n \mid n \in \mathbb{N}\}$, so $S \models \{\sigma\}\, X\, \{\sigma\}$. Suppose that G is complete, and thus

$$Tr_S \vdash \{\sigma\}\, X\, \{\sigma\}.$$

We may assume that the last two rules applied are V respectively IV (rule IV is the only rule not adding complexity to the process expression involved). So there must be α, β such that

$$Tr_S, \{\alpha\}\, x\, \{\beta\} \vdash \{\alpha\}\, \tau b x c + d\, \{\beta\} \tag{1}$$

$$\sigma \rightarrow \alpha, \beta \rightarrow \sigma \in Tr_S. \tag{2}$$

Now (1) implies that $Tr_S \vdash \{\alpha\}\, d\, \{\beta\}$ and by (2) we derive $Tr_S \vdash \{\sigma\}\, d\, \{\beta\}$. By the soundness of G we conclude that $\sigma \rightarrow \beta \in Tr_S$, so by (2) we have

$$\beta \leftrightarrow \sigma \in Tr_S. \tag{3}$$

Also $Tr_S, \{\alpha\}\, x\, \{\beta\} \vdash \{\alpha\}\, \tau bxc\, \{\beta\}$, so there must be γ_1, γ_2 such that

$$Tr_S, \{\alpha\}\, x\, \{\beta\} \vdash \big\{\{\alpha\}\, \tau b\, \{\gamma_1\}\,,\ \{\gamma_1\}\, x\, \{\gamma_2\}\,,\ \{\gamma_2\}\, c\, \{\beta\}\big\}.$$

Because $\{\gamma_1\}\, x\, \{\gamma_2\}$ is derivable from $\{\alpha\}\, x\, \{\beta\}$ we have

$$\beta \rightarrow \gamma_2 \in Tr_S \tag{4}$$

and since $\{\gamma_2\} c \{\beta\}$ is derivable as well we conclude by the soundness of G and (3) that $\gamma_2 \rightarrow \sigma' \in Tr_S$, so by (4) we have

$$\beta \rightarrow \sigma' \in Tr_S. \tag{5}$$

Now (3) and (5) are contradictory, so the proof system G is incomplete with respect to all guarded systems.

6 Restricted guarded systems and Hoare's logic

In this section we present a final proof system H by extending G with a rule of substitution, which will be proved sound and complete. This means that from now on we will look at partial correctness assertions about processes *modulo* derivability (in section 4 we still were able to derive a proof rule 'substitution'). A crucial result with respect to this section states that any guarded system can be proved equal to a semi linear system.

6.1 The proof system H

Definition 6.1.1 *A* restricted guarded system $E = \{x = t_x \,|\, x \in V_E\}$ *is a guarded system in which the expressions t_x do not contain guarded systems.*

We specify the process domain $\mathcal{P}_r$ over $\mathrm{BPA}_{\delta\tau} + \mathrm{REC}$ as follows:

- $a \in \mathcal{P}_r$ if $a \in A_{\delta\tau}$
- $<x|E> \in \mathcal{P}_r$ if $E = \{x = t_x \,|\, x \in V_E\}$ is a restricted guarded system
- $pq \in \mathcal{P}_r$ if $p, q \in \mathcal{P}_r$
- $p + q \in \mathcal{P}_r$ if $p, q \in \mathcal{P}_r$
- $pq \in \mathcal{P}_r$ if $p, q \in \mathcal{P}_r$

The notation $\mathcal{P}_r$ is used because H will also be discussed with respect to a larger process domain $\mathcal{P}_H$ in which 'nested recursion' is allowed (see section 7). We extend the proof system G to H by redefining rule V (*recursion*) as referring to a *restricted guarded* system $E = \{x = t_x \,|\, x \in V_E\}$ and adding a rule of 'substitution':

$$\text{VI} \quad \textit{substitution} \quad \frac{\{\alpha\}\, p\, \{\beta\} \qquad T \vdash p = q}{\{\alpha\}\, q\, \{\beta\}}$$

where T is short for $\mathrm{BPA}_{\delta\tau} + \mathrm{REC} + \mathrm{PR} + \mathrm{B} + \mathrm{AIP}^-$.

6.2 H is complete

We use the proof system K (see section 5.2) to show that H is sound, and therefore extend K to K^* by redefining rule V* as referring to a restricted guarded system $E = \{x = t_x \,|\, x \in V_E\}$, and adding the rule

$$\text{VII}^* \quad \frac{\Phi \longrightarrow \{\{\alpha\}\, p\, \{\beta\}\} \qquad T \vdash p = q}{\Phi \longrightarrow \{\{\alpha\}\, q\, \{\beta\}\}}$$

Lemma 6.2.1 *The proof system K^* is sound.*

Proof. The soundness of rule VII* follows easy by corollary 3.3.6. We only prove the soundness of rule V*. Let S be fixed and $E = \{x = t_x \,|\, x \in V_E\}$ a restricted guarded system. Suppose

$$S \models \big\{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\big\} \longrightarrow \big\{\{\alpha_x\}\, t_x\, \{\beta_x\} \mid x \in V_E\big\}.$$

If we just prove

$$\forall n \in \mathbb{N}\big(S \models \big\{\{\alpha_x\}\, \pi_n(<x|E>)\, \{\beta_x\} \mid x \in V_E\big\}\big)$$

then by the proof of lemma 5.2.2 we are done. Let $E' = \{x = t'_x \,|\, x \in V_E\}$ be constructed by removing the brackets in the expressions t_x, using the axioms A4 and CTR3 (see table 3). It follows easy that

$$\frac{S \models \big\{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\big\} \longrightarrow \big\{\{\alpha_x\}\, t_x\, \{\beta_x\} \mid x \in V_E\big\}}{S \models \big\{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\big\} \longrightarrow \big\{\{\alpha_x\}\, t'_x\, \{\beta_x\} \mid x \in V_E\big\}}$$

and $S \models <x|E> =_{pc} <x|E'>$ for all $x \in V_E$ by corollary 3.3.6. We show

$$\forall n \in \mathbb{N}\big(S \models \big\{\{\alpha_x\}\, \pi_n(<x|E'>)\, \{\beta_x\} \mid x \in V_E\big\}\big).$$

$n = 0:$ Substituting δ for all variables of V_E we have by supposition that

$$S \models \big\{\{\alpha_x\}\, t'_x(\,\overline{\delta}\,)\, \{\beta_x\} \mid x \in V_E\big\}.$$

Fix some $x_0 \in V_E$ and consider all summands of t'_{x_0} not containing variables of V_E, say r_j $(j = 1 \cdots n, n \geq 0)$. By corollary 3.3.6 we have that $S \models t'_{x_0}(\,\overline{\delta}\,) =_{pc} \delta + \sum r_j$, so $S \models \{\alpha_{x_0}\}\, \delta + \sum r_j\, \{\beta_{x_0}\}$. It is not difficult to see that $S \models \{\alpha_{x_0}\}\, \pi_0(\delta + \sum r_j)\, \{\beta_{x_0}\}$. Now $T \vdash \pi_0(<x_0\,|\,E'>) = \pi_0(\delta + \sum r_j)$, for E' is a guarded system, so $S \models \{\alpha_{x_0}\}\, \pi_0(<t_{x_0}\,|\,E'>$ $)\, \{\beta_{x_0}\}$ by corollary 3.3.6. We conclude

$$S \models \big\{\{\alpha_x\}\, \pi_0(<x|E'>)\, \{\beta_x\} \mid x \in V_E\big\}.$$

$n + 1:$ Assume that for some $x_0 \in V_E$ we have $\alpha_{x_0}(s)$ and

$$(\pi_{n+1}(<x_0\,|\,E'>), s) \overset{\sigma}{\twoheadrightarrow} (\surd, s').$$

Let $t'_{x_0} \equiv \sum p_i y_i q_i \;\; + \sum r_j$, where the y_i are in V_E, the expressions q_i may contain variables of V_E and the expressions p_i and r_j do not contain variables of V_E. Now we have that

$$(\pi_{n+1}(r_{j_0}), s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')$$

for a summand r_{j_0} of t'_{x_0}, or

$$(\pi_{n+1}(<p_{i_0} y_{i_0} q_{i_0}\,|\,E'>), s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')$$

for a summand $p_{i_0}y_{i_0}q_{i_0}$ of t'_{x_0}. In the last case we have

$$(\pi_{n+1}(<p_{i_0}y_{i0}q_{i_0}\,|\,E'>), s) \overset{\sigma_1}{\twoheadrightarrow} (\pi_m(<y_{i_0}q_{i_0}\,|\,E'>), s'') \overset{\sigma_2}{\twoheadrightarrow} (\surd, s')$$

for some $s'' \in S$, such that $\sigma_1\sigma_2 \equiv \sigma$ and by the guardedness of E' also $m \leq n$. In both cases it can be shown that

$$(t'_{x_0}(\,\overline{\pi_n(<x\,|\,E'>)}\,), s) \overset{\sigma}{\twoheadrightarrow} (\surd, s')$$

using

1. If $(\pi_n(pq), s) \overset{\rho}{\twoheadrightarrow} (\surd, s')$, then $(\pi_n(p)\pi_n(q), s) \overset{\rho}{\twoheadrightarrow} (\surd, s')$,
   if $(\pi_n(p), s) \overset{\rho}{\twoheadrightarrow} (\surd, s')$, then $(p, s) \overset{\rho}{\twoheadrightarrow} (\surd, s')$ and $\forall k \in \mathbb{N}\big((\pi_{n+k}(p), s) \overset{\rho}{\twoheadrightarrow} (\surd, s')\big)$.
2. If $(p, s) \overset{\rho}{\twoheadrightarrow} (\surd, s') \implies (q, s) \overset{\rho}{\twoheadrightarrow} (\surd, s')$, then for all $r \in \mathcal{P}_r$:
   $(rp, \hat{s}) \overset{\nu}{\twoheadrightarrow} (\surd, \hat{s}') \implies (rq, \hat{s}) \overset{\nu}{\twoheadrightarrow} (\surd, \hat{s}')$.

By supposition and the induction hypothesis we conclude $\beta_{x_0}(s')$ and thus

$$S \models \big\{\{\alpha_x\}\,\pi_{n+1}(<x\,|\,E'>)\,\{\beta_x\} \mid x \in V_E\big\}.$$

□

Now the soundness of H follows easy:

Lemma 6.2.2 *If* $\Gamma, \Phi \vdash^H \Psi$, *then* $\Gamma \vdash^{K^*} \Phi \longrightarrow \Psi$.

Proof. As the proof of lemma 5.2.3.

Lemma 6.2.3 *The proof system H is sound.*

Proof. See the preceding lemma and the proof of lemma 5.2.4.

We now consider the issue of the completeness of H.

Lemma 6.2.4 *If* $E = \{x = t_x \,|\, x \in V_E\}$ *is a restricted guarded system, and we have for some* S *and* $x_0 \in V_E$ *that*

$$S \models \{\alpha\}<x_0\,|\,E>\{\beta\}$$

then also

$$Tr_S \vdash \{\alpha\}<x_0\,|\,E>\{\beta\}.$$

Proof. Construct a *semi linear* system $E' = \{x = t'_x \,|\, x \in V_{E'}\}$ such that

$$T \vdash <x_0\,|\,E> = <y_0\,|\,E'>$$

for some $y_0 \in V_{E'}$. By corollary 3.3.6 we have that $S \models \{\alpha\}<y_0\,|\,E'>\{\beta\}$, and by the completeness of G we conclude $Tr_S \vdash^G <y_0\,|\,E'>$, so by VI we derive $Tr_S \vdash^H <x_0\,|\,E>$.

A possible construction can be found in [10], we just present an example: Let $b, c, d \in A$ and $E = \{X = \tau bXc + d\}$ and take $E' = \{Y_n = \tau bY_{n+1} + dc^n \,|\, n \in \mathbb{N}\}$. Now $T \vdash <X\,|\,E> = <Y_0\,|\,E'>$. (We can prove that $\forall n, k(\pi_n(Xc^k) = \pi_n(Y_k)$ by induction on n, and then apply AIP$^-$.) □

We conclude section 6 with the following result:

Theorem 6.2.5 *The proof system H is complete.*

Proof. The soundness of H is proved in lemma 6.2.2. As for the other side of the question we again refer to the proof of theorem 4.2.2. There is one more basic clause of $\mathcal{P}_r$ to inspect: $p \equiv <x\,|\,E>$ with $E = \{x = t_x \,|\, x \in V_E\}$ a restricted guarded system. This has just been done in lemma 6.2.4. □

I	*axioms* $(a \in A_{\delta\tau})$	$\dfrac{\big(\neg stop(a,x) \wedge \alpha(x)\big) \;\rightarrow\; \beta(effect(x,a))}{\{\alpha\}\, a\, \{\beta\}}$
II	*alternative composition*	$\dfrac{\{\alpha\}\, p\, \{\beta\} \quad \{\alpha\}\, q\, \{\beta\}}{\{\alpha\}\, p+q\, \{\beta\}}$
III	*sequential composition*	$\dfrac{\{\alpha\}\, p\, \{\beta\} \quad \{\beta\}\, q\, \{\gamma\}}{\{\alpha\}\, pq\, \{\beta\}}$
IV	*consequence*	$\dfrac{\alpha \rightarrow \alpha' \quad \{\alpha'\}\, p\, \{\beta'\} \quad \beta' \rightarrow \beta}{\{\alpha\}\, p\, \{\beta\}}$
V	*recursion*	*If* $E = \{x = t_x \,\vert\, x \in V_E\}$ *is a guarded system, then* $\dfrac{\big\{\{\alpha_x\}\, x\, \{\beta_x\} \mid x \in V_E\big\} \;\vdash\; \big\{\{\alpha_x\}\, t_x\, \{\beta_x\} \mid x \in V_E\big\}}{\big\{\{\alpha_x\} <x \vert E> \{\beta_x\} \mid x \in V_E\big\}}$
VI	*substitution*	$\dfrac{\{\alpha\}\, p\, \{\beta\} \qquad T \vdash p = q}{\{\alpha\}\, q\, \{\beta\}}$

Table 7: The proof system H

7 Guarded systems and Hoare's logic

In this section we finally consider H with respect to a process domain containing all guarded systems. We show that H is complete on this domain.

7.1 The proof system H

We specify the process domain $\mathcal{P}_H$ over BPA$_{\delta\tau}$ + REC as follows:

- $a \in \mathcal{P}_H$ if $a \in A_{\delta\tau}$
- $<x \vert E> \in \mathcal{P}_H$ if $E = \{x = t_x \,\vert\, x \in V_E\}$ is a guarded system
- $p + q \in \mathcal{P}_H$ if $p, q \in \mathcal{P}_H$
- $pq \in \mathcal{P}_H$ if $p, q \in \mathcal{P}_H$

For the sake of completeness, we sum up H as a whole in table 7.

7.2 H is complete

Before we prove the completeness of H, we introduce the means to remove nested recursion. We first give an example.

Example. Consider the guarded systems $E_x = \{x = ax + b\}, E_y = \{y = c{<}x\,|\,E_x{>}y + d{<}x\,|\,E_x{>}\}$ and $E_z = \{z = {<}x\,|\,E_x{>}b + c{<}y\,|\,E_y{>} + azc\}$. Let $p \equiv {<}z\,|\,E_z{>}$. We construct $p^+ \equiv {<}z\,|\,E_z^+{>}$ by adding the equation $x = ax + b$ to E_z and replacing the summand ${<}x|E_x{>}b$ in E_z by $(ax + b)b$. Call the resulting guarded system E^+. We can prove that $T \vdash p = p^+$ and apparently p^+ is an expression which is 'more simple' in terms of nested recursion: We may represent p as

$$<z\,|\,\{\quad z = {<}x|E_x{>}b + c{<}y\,|\,\{y = {\scriptstyle c<x\,|\,E_x>y + d<x\,|\,E_x>}\}{>} \ + azc \quad \}>,$$

whereas p^+ can be represented as

$$\begin{array}{lll} <z\,|\,\{ & z = (ax + b)b + c{<}y\,|\,\{y = {\scriptstyle c<x\,|\,E_x>y + d<x\,|\,E_x>}\}{>} \ + azc & \\ & x = ax + b & \}>. \end{array}$$

We formalize this idea as follows:

Definition 7.2.1 *Let $E = \{x = t_x \,|\, x \in V_E\}$ be a guarded system and $x_0 \in V_E$. For any guardedly specified process expression occurring in E we define its* nesting level *as the depth of the nesting of this occurrence in E. The* nesting number *of ${<}x_0\,|\,E{>}$ is the sum of the nesting levels of all guardedly specified process expressions occurring in E. Let $p \in P_H$. The nesting number of p is the sum of all nesting numbers of guardedly specified process expressions occurring in p.*

In the example above we see that ${<}x\,|\,E_x{>}$ occurs twice in E_z with nesting level 2 (in the summand $c{<}y\,|\,E_y{>}$), and once in E_z with nesting level 1. The nesting number of p is 6, the nesting number of p^+ is 5.

Lemma 7.2.2 *The proof system H is sound.*

Proof. Suppose $Tr_S \vdash \{\alpha\}\, p\, \{\beta\}$ for some p. We prove by induction on the nesting number of p, say n, that $S \models \{\alpha\}\, p\, \{\beta\}$.

$n = 0$: Now p denotes an element of P_r. Because we have no means in H to decrease the nesting number of p, we may *derive* $\{\alpha\}\, p\, \{\beta\}$ in H with respect to P_r (that is, by restricting the applicability of V to guarded systems having nesting number 0), so by the result of section 6 we are done.

$n + 1$: Fix an occurrence of a guardedly specified process expression in p with nesting number ≥ 1, say ${<}x_0\,|\,E{>}$ with $E = \{x = t_x \,|\, x \in V_E\}$ (by the induction hypothesis we can find such an occurrence). Select a guardedly specified process expression with nesting level 1, say ${<}y_0\,|\,E'{>}$ with $E' = \{x = t'_x \,|\, x \in V_{E'}\}$. Now suppose $V_E \cap V_{E'} = \emptyset$ (this is not a restriction, for we may rename the variables in E'). Replace the *selected* occurrence of ${<}y_0\,|\,E'{>}$ in E by t'_{y_0} (of course there may be more occurrences of ${<}y_0\,|\,E'{>}$ in E) and let $x_0 = t^+_{x_0}$ denote the resulting equation. Further add the equations of E' to E, thus constructing $E^+ = \{x = t^+_x \,|\, x \in V_E \cup V_{E'}\}$ where $t^+_x = t_x$ for all $x \in V_E - \{x_0\}$ and $t^+_x = t'_x$ for all $x \in V_{E'}$. The system E^+ is again a guarded system and with the principle AIP$^-$ we can prove $T \vdash p = p^+$, where p^+ is obtained by replacing the fixed occurrence of ${<}x_0\,|\,E{>}$ by ${<}x_0\,|\,E^+{>}$ in p. Applying rule VI we derive $Tr_S \vdash \{\alpha\}\, p^+\, \{\beta\}$, and since the nesting number of p^+ is n, we have by the induction hypothesis that $S \models \{\alpha\}\, p^+\, \{\beta\}$. By corollary 3.3.6 we conclude $S \models \{\alpha\}\, p\, \{\beta\}$. □

Theorem 7.2.3 *The proof system H is complete.*

Proof. The soundness of H is proved in lemma 7.2.2. For the converse we only check the basic clause $p \equiv \,<x \,|\, E>$ with $E = \{x = t_x \,|\, x \in V_E\}$ a guarded system. As suggested by definition 7.2.1 just replace all nested recursions in the expressions t_x by fresh variables not in V_E and add all belonging (renamed) equations to E, in this way constructing $E' = \{x = t_x \,|\, x \in V_E \cup V_{E'}\}$. It follows easy that $T \vdash \,<x|E> \,=\, <x|E'>$ for all $x \in V_E$. Now if $\mathcal{S} \models \{\alpha\} <x \,|\, E> \{\beta\}$ for some $x \in V_E$, then we have $\mathcal{S} \models \{\alpha\} <x|E'> \{\beta\}$. By completeness of H with respect to $\mathcal{P}_r$ we conclude $Tr_{\mathcal{S}} \vdash \{\alpha\} <x \,|\, E'> \{\beta\}$, and thus $Tr_{\mathcal{S}} \vdash \{\alpha\} <x \,|\, E> \{\beta\}$ by rule VI. □

8 Final remarks

8.1 The projection operator

One may wonder why the projection operators π_n were not included in the final proof system H. A simple proof rule concerning the introduction of these operators like

$$\frac{\{\alpha\}\, p\, \{\beta\}}{\{\alpha\}\, \pi_n(p)\, \{\beta\}}\, n \in \mathbb{N}$$

is obviously sound, but we lose completeness: All structures $\mathcal{S}$ satisfy $\mathcal{S} \models \{\alpha\}\, \pi_0(p)\, \{\beta\}$ for all p and α, β. However, via the rule of substitution we are able to deal with partial correctness assertions containing these operators, because for any $p \in \mathcal{P}$ there is a $q \in \mathcal{P}_H$ (and thus π_n-free) such that $T \vdash p = q$ and with corollary 3.3.6 we can transfer results back to $\mathcal{P}$.

8.2 The merge operator

A burning question is of course which proof rules concerning the ACP$_\tau$ framework can be defined in this set-up[4]. Referring to the action rules defined in [6], it is not difficult to think of effect rules concerning the ACP$_\tau$-operators. As an example we may define effect rules introducing the merge operator $\|$ as follows:

$$\|: \quad \frac{(x,s) \xrightarrow{a} (x',s')}{(x \,\|\, y, s) \xrightarrow{a} (x' \,\|\, y, s')} \qquad \frac{(x,s) \xrightarrow{a} (\surd, s')}{(x \,\|\, y, s) \xrightarrow{a} (y, s')}$$

$$\frac{(y,s) \xrightarrow{a} (y',s')}{(x \,\|\, y, s) \xrightarrow{a} (x \,\|\, y', s')} \qquad \frac{(y,s) \xrightarrow{a} (\surd, s')}{(x \,\|\, y, s) \xrightarrow{a} (x, s')}$$

$$\frac{(x,s) \xrightarrow{a} (x',s') \quad (y,s) \xrightarrow{b} (y',s')}{(x \,\|\, y, s) \xrightarrow{a|b} (x' \,\|\, y', s')} \qquad \frac{(x,s) \xrightarrow{a} (\surd,s') \quad (y,s) \xrightarrow{b} (y',s')}{(x \,\|\, y, s) \xrightarrow{a|b} (y', s')} \qquad (\text{if } a|b \in A_\tau)$$

$$\frac{(x,s) \xrightarrow{a} (x',s') \quad (y,s) \xrightarrow{b} (\surd,s')}{(x \,\|\, y, s) \xrightarrow{a|b} (x', s')} \qquad \frac{(x,s) \xrightarrow{a} (\surd,s') \quad (y,s) \xrightarrow{a} (\surd,s')}{(x \,\|\, y, s) \xrightarrow{a|b} (\surd, s')} \qquad (\text{if } a|b \in A_\tau)$$

A proof rule introducing the merge operator of the form

$$\frac{\{\alpha_1\}\, p\, \{\beta_1\} \quad \{\alpha_2\}\, p\, \{\beta_2\}}{\{\alpha_3\}\, p \,\|\, q\, \{\beta_3\}}$$

[4] The ACP$_\tau$ axioms can be found in [5,6].

where the α_i, β_i may be somehow related cannot be sound: As an example consider the process $ab \parallel c$ with $a, b, c \in A$, which may be represented as $a(bc + cb) + (a|c)b$. Now a possible course of action for this process is to execute a, c, b, but the premisses of this 'merge rule' do not contain any information concerning this possibility, since only effects of the execution of a followed by the execution of b are considered.

An alternative could be to define a set $\texttt{effectless}(A) \subseteq A$, containing the atomic actions which are inert with respect to the function *effect*. Now we may extend H with a proof rule

$$\text{VII} \quad \textit{merge} \quad \frac{\{\alpha\}\, p\, \{\beta\} \quad \{\alpha'\}\, q\, \{\beta'\}}{\{\alpha \wedge \alpha'\}\, p \parallel q\, \{\beta \vee \beta'\}} \ \texttt{alphabet}(p) \subseteq \texttt{effectless}(A) \supseteq \texttt{alphabet}(q)$$

where $\texttt{alphabet}(p)$ returns the set of atomic actions occurring in p. Provided that all communications are in $\texttt{effectless}(A)$, we can prove a completeness result for such an extension of H (using effect rules for the $|$ -operator as suggested by [6]).

Now consider the following effect rules, introducing the abstraction operator τ_I:

$$\tau_I : \quad \frac{(x,s) \xrightarrow{a(s)} (y,s')}{(\tau_I(x),s) \xrightarrow{a(s)} (\tau_I(y),s')} \quad \frac{(x,s) \xrightarrow{a(s)} (\surd,s')}{(\tau_I(x),s) \xrightarrow{a(s)} (\surd,s')} \quad (\text{if } a \notin I)$$

$$\frac{(x,s) \xrightarrow{a(s)} (y,s')}{(\tau_I(x),s) \xrightarrow{\tau} (\tau_I(y),s')} \quad \frac{(x,s) \xrightarrow{a(s)} (\surd,s')}{(\tau_I(x),s) \xrightarrow{\tau} (\surd,s')} \quad (\text{if } a \in I)$$

Defining $\texttt{inert}(A) \subseteq \texttt{effectless}(A)$ as the set of atomic actions being inert, we can even further extend H without losing completeness by adding a proof rule

$$\text{VIII} \quad \textit{abstraction} \quad \frac{\{\alpha\}\, p\, \{\beta\}}{\{\alpha\}\, \tau_I(p)\, \{\beta\}} \ I \subseteq \texttt{inert}(A)$$

because $\tau_I(p)$ and p then satisfy exactly the same effect reductions.

Finally observe that concerning the encapsulation operator ∂_H we encounter the same problems as discussed in remark 1.

8.3 The state operator

The reader acquainted with process algebra may wonder why we used 'state labelled process expressions' instead of the state operator λ_s (see eg. [2]). We argue that all axioms used, if any, have to be sound. Since the state operator is defined axiomatically, it may relate processes having *different* initial states. As an example take $s \neq s'$, $a(s) = b(s')$ and $s(a) = s'(b)$ and derive $\lambda_s(ax) = \lambda_{s'}(bx)$. Now there is no appropriate semantical view on the identification of processes having different initial states.

Acknowledgements

I would like to thank Jan Bergstra for leading me to this subject, and for his constructive help in writing this paper. For critical remarks, discussions and corrections I also thank Jos Baeten, Henk Goeman and Frits Vaandrager.

References

[1] K.R. APT, *Ten Years of Hoare's Logic: A Survey — Part 1*, in: ACM Transactions on Programming Languages and Systems, Vol. 3, No. 4, 1981, pp. 431-483.

[2] J.C.M. BAETEN, J.A. BERGSTRA, *Global Renaming Operators over Concrete Process Algebra*, in: Inf. & Comp. 78(3), 1988, pp. 205-245.

[3] J.C.M. BAETEN, J.A. BERGSTRA, *Recursive Process Definitions with the State Operator*, in: Proceedings Computing Science in the Netherlands (SION), 1988, pp. 279-294.

[4] J.W DE BAKKER, J.N. KOK, J.-J.CH. MEYER, E.-R. OLDEROG, J.I. ZUCKER, *Contrasting themes in the semantics of imperative concurrency*, in: Current Trends in Concurrency (J.W. de Bakker, W.P. de Roever, G. Rozenberg, eds.), LNCS 224, Springer-Verlag, 1986, pp. 51-121.

[5] J.A. BERGSTRA, J.W. KLOP, *Process Algebra: Specification and Verification in Bisimulation Semantics*, in: Mathematics and Computer Science II, CWI monograph 4 (M. Hazewinkel, J.K. Lenstra, L.G.L.T. Meertens, eds.), North-Holland, Amsterdam, 1986, pp. 61-94.

[6] R.J. VAN GLABBEEK, *Bounded Nondeterminism and the Approximation Induction Principle in Process Algebra*, in: Proceedings STACS 87 (F.J. Brandenburg, G. Vidal-Naquet, M. Wirsing, eds.), LNCS 247, Springer-Verlag, 1987, pp. 336-347.

[7] J.F. GROOTE, F.W. VAANDRAGER, *Structured Operational Semantics and Bisimulation as a Congruence*, Report CS-R8845, Centrum voor Wiskunde en Informatica, Amsterdam, 1988.

[8] D.M.R. PARK, *Concurrency and automata on infinite sequences*, in: Proceedings 5th GI Conference (P. Deussen, ed.), LNCS 104, Springer-Verlag, 1981, pp. 167-183.

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

[10] W.P. WEIJLAND, *Synchronous and Asynchronous Processes*, to appear as: Report CS-R88.., Centrum voor Wiskunde en Informatica, Amsterdam.