An Operator Definition Principle (for Process Algebras)

C. Verhoef

Programming Research Group
University of Amsterdam
Kruislaan 403
1098 SJ Amsterdam
e-mail: chris@fwi.uva.nl

Abstract. In a first attempt to bring some structure in the use of linear unary operators in process algebra, we propose two principles, in order to obtain a uniform approach for the introduction of these operators. For this we will use the notion of a linear functional specification that consists of functional equations and boundary conditions; the operator definition principle states that such a system has a solution. Furthermore, we define an operator specification principle; this principle states that a linear functional specification has at most one solution. Examples to demonstrate the usage of the two principles are given. We can think of the use of auxiliary operators in verifications, specifications, or a combination of both. Although these specific operators are, in general, not usable in another context, there is a need for auxiliary operators that can be defined as we wish. Moreover, we will need even more complicated auxiliary linear unary operators in the future, and therefore, a more sophisticated definition of a linear functional specification. Suggestions to generalize this concept are given, too.

Key Words & Phrases: *Process Algebras, Algebra of Communicating Processes (with abstraction), Linear Unary Operators, Concurrency, Operator Definition Principle, Operator Specification Principle, Unification of Process Algebras.*

1980 Mathematics Subject Classification (1985 Revision): *68Q10, 68Q40, 68Q45, 68Q99.*
CR Categories: *F.1.2, F.3.1.*

Note: *Partial support received from the European Communities under ESPRIT project no 3006, CONCUR.*

Contents

1. Introduction

We will consider in this paper the algebra of communicating processes with abstraction and linear unary operators ($ACP_{\tau,u}$). We will introduce $ACP_{\tau,u}$ as an extension of ACP_τ, the algebra of communicating processes with abstraction, (see [**6**]). Hereinafter, we will give an overview of what can be expected in the subsequent sections and we will give some motivation for the theory that will be presented in this paper.

In section 2 we will give the signature and the axioms of $ACP_{\tau,u}$. Furthermore, we will formulate the operator definition principle and the operator specification principle in terms of solutions for linear functional specifications. In $ACP_{\tau,u}$, we will have two sorts: the sort of processes and the sort of linear unary operators. In the sort of processes we will have the "usual" constants: the atomic actions and the special constants. In the sort of linear unary operators we will have the projection operators, the encapsulation operator and the abstraction operator. It can be found that, with the aid of this theory, we can introduce the latter two operators. We still added them to the set of constants, since they are *not* auxiliary: with these operators we want to formulate other axioms. We can mention here $KFAR_n$ ([**10**]) and conditional axioms ([**3**]). The same yields for the projection operators: we formulate the approximation induction principle with them. So, even in a generalized version of this theory in which we can specify the projection operators (see section 8), we want to have these operators as constants. We will redefine certain items that are already known in ACP_τ, for instance, the notion of a guard and of a guarded recursive specification. This will be done in definition (2.15). We will give a motivation on the alterations that we made into these notions. We will introduce two more principles: the recursive definition principle (*RDP*) and the recursive specification principle (*RSP*). These principles can be found in [**7**]. We will also introduce the approximation induction principle (*AIP*) that can be found in [**7**], too.

In section 3 we will prove the termination of the system that we introduced in section 2 by means of a method that is called the recursive path ordering. In [**6**] this method is used to prove the termination of ACP_τ. We will use the same method to prove the termination of $ACP_{\tau,u}$. We will explain this method and we will give all the necessary definitions. Thereinafter, we will prove an elimination result, which states that every closed $ACP_{\tau,u}$-term can be rewritten into a closed $BPA_{\delta,\tau}$-term.

In section 4 we will introduce some more concepts that are necessary in order to state general theorems concerning linear unary operators, which can be defined with the aid of linear functional specifications. In this section it will become clear that with this theory we do not longer have to prove for each auxiliary linear unary operator separately that it has certain properties: it will be sufficient to verify that this operator satisfies the conditions stated in these general theorems in order to know that it has the desired properties. We attempted to handle a great variety of subjects. But it will be far from complete. Many questions could be asked, and answered, on the subject of linear unary operators. For instance, we want to know what processes are fixed points of a linear unary operator, or: "When do linear unary operators commute?" Just to mention a few.

In section 5 we will construct a model for ACP_u. This is the axiom system $ACP_{\tau,u}$, but without abstraction. We did this because we wanted to construct what is called the standard model of process algebra, that is, the inverse or projective limit model. Moreover, it is a well-known fact that the combination of the projective limit model with the concept of abstraction is very problematic. We will prove that a desired property (such as *RDP*, or *OSP*) is valid for the elements of the inverse system and we use this to prove that it is preserved by taking the inverse limit. We think that some of these proofs can be shortened by using so-called preservation theorems on inverse limits, but since the theory on preservation theorems is only developed for single-sorted algebras, we will give direct proofs.

In section 6 we will apply the theory by giving some examples of verifications and specifications with the aid of auxiliary linear unary operators. We will also see that this theory is used to prove certain properties of operators that we already know. For instance, if we have the abstraction operator τ_I (this operator renames all the atomic actions that are in the subset $I \subseteq A$ into the silent step τ), then we immediately "feel" that $\tau_I \circ \tau_I$ must be τ_I. With the aid of *OSP* it is, actually, very trivial to prove this. It turns out that, with the aid of the general theorems in section 4, it will be sufficient to verify that τ_I^2 and τ_I are the same on the set of atomic actions in order to conclude that they are equal.

In section 7 we will give some suggestions how we can generalize this theory in order to be able to describe more (auxiliary) linear unary operators. Such as the projection operators or the generalized state operator, just to mention a few. We will also give a short example of the increase of the proving power of the generalized theory.

In section 8 we will emphasize that this theory can be the beginning of the unification of all the theories, built up from *ACP* or ACP_τ and, in addition, a number of auxiliary linear unary operators, by discussing the results of section 6 in more detail.

In section 9 we will catalogue all the references we are going to make throughout the paper. We will try to refer to the first source in which a concept can be found.

Finally, we will include a glossary in section 10.

Acknowledgement

The author would like to thank J. A. Bergstra for his comments and suggestions to improve the draft versions of this paper.

2. Definitions

In this section we will consider the algebra of communicating processes with abstraction and linear unary operators. We will use the following notation for this system: $ACP_{\tau,u}$. First, we will give a graphical representation of the signature of $ACP_{\tau,u}$ in figure 1. Subsequently, we will enumerate the same signature in a more textual way.

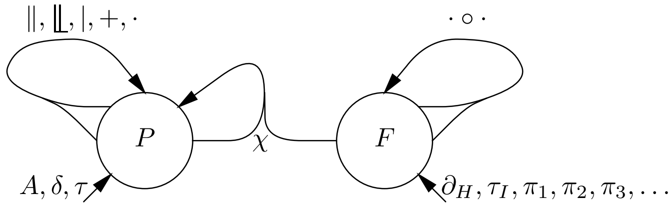

Figure 1. Graphical representation of the signature.

From this "pair of spectacles" it is not clear what the order of the arguments in the function χ is. We will also give the naming of the symbols, used in figure 1. First, we consider the sort P. We have a set of constants A, or atomic actions. We have also two special constants in P: δ or deadlock, and τ or silent step. Consider the binary operators (they are all infix operators):

merge:	$\parallel : P \times P \longrightarrow P,$
left-merge:	$⌊\!⌊ : P \times P \longrightarrow P,$
communication-merge:	$\mid : P \times P \longrightarrow P,$
sequential composition:	$\cdot : P \times P \longrightarrow P,$
alternative composition:	$+ : P \times P \longrightarrow P.$

Now we consider the sort F. For each $I \subseteq A$ we have a constant $\tau_I \in F$, which is called the abstraction operator. For each $H \subseteq A$ we have a constant ∂_H in F. This constant is called the encapsulation operator. For each $n \geq 1$ we have a constant $\pi_n \in F$. The constant π_n is called the (nth) projection operator. There is one binary operation with both arguments of sort F; it is the composition of functions: $\circ : F \times F \longrightarrow F$. Finally, we have a binary operation $\chi : F \times P \longrightarrow P$. This operator can be called the apply function. This concludes our discussion on the signature of $ACP_{\tau,u}$. Hereinafter, we will give the axiom system of $ACP_{\tau,u}$ in table 1 on page 6. In this table we use the following notational conventions: a, b, c are atomic actions or δ (we abbreviate $A_\delta = A \cup \{\delta\}$); x, y, z are processes; γ is a special constant (we use $C = \{\delta, \tau\}$ for the set of special constants); $n \geq 1$, and finally, f, g and h are linear unary operators.

At this point we will formulate an extensionality axiom which states: functions that behave the same on all processes are indeed the same. We will not use the axiom in this paper but we added it since there is no reason not to have it.

Axiom (2.1) *Extensionality*

Let f and g be linear unary operators. If for all processes $x \in P : \chi(f, x) = \chi(g, x)$, then $f = g$. We will use the abbreviation *EA* for this axiom.

Definition (2.2)

Let N be a finite set of function names. A linear functional specification $E(N)$ for N is a set of the following form:

$$E(N) = \left\{\, r_{n,a} \mid n \in N, a \in A \,\right\} \cup \left\{\, e_{n,a} \mid n \in N, a \in A \,\right\}. \tag{1}$$

Both $r_{n,a}$ and $e_{n,a}$ are equations. Let $a \in A$ and $n \in N$ be fixed. Then we define the two equations $r_{n,a}$ and $e_{n,a}$ for this particular pair (n, a). The first equation $r_{n,a}$ is called a boundary condition and has the following form: there is an element $b \in A \cup C$ such that

$$r_{n,a} \equiv \chi(n, a) = b.$$

The second equation $e_{n,a}$ is called a (linear) functional equation and it is of the following form:

$$e_{n,a} \equiv \chi(n, a \cdot x) = b \cdot \chi(m, x) \quad \text{for one } m \in N.$$

Examples of linear functional specifications can be found in remarks (2.11) and (2.19). Furthermore, if the set of function names contains only one element , say n, we will omit the braces in the notation of (1): we will write $E(n)$ instead of $E(\{n\})$.

$x + y = y + x$	A1	$x \cdot \tau = x$	T1
$x + (y + z) = (x + y) + z$	A2	$\tau \cdot x + x = \tau \cdot x$	T2
$x + x = x$	A3	$a \cdot (\tau \cdot x + y) = a \cdot (\tau \cdot x + y) + a \cdot x$	T3
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4		
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5	$\tau \mathbin{\|\!\lfloor} x = \tau \cdot x$	TM1
$x + \delta = x$	A6	$(\tau \cdot x) \mathbin{\|\!\lfloor} y = \tau \cdot (x \parallel y)$	TM2
$\delta \cdot x = \delta$	A7	$\tau \mid x = \delta$	TC1
		$x \mid \tau = \delta$	TC2
$a \mid b = b \mid a$	C1	$(\tau \cdot x) \mid y = x \mid y$	TC3
$(a \mid b) \mid c = a \mid (b \mid c)$	C2	$x \mid (\tau \cdot y) = x \mid y$	TC4
$\delta \mid a = \delta$	C3		
		$\chi(\tau_I, a) = a, \quad \text{if } a \notin I$	TI1
$x \parallel y = x \mathbin{\|\!\lfloor} y + y \mathbin{\|\!\lfloor} x + x \mid y$	CM1	$\chi(\tau_I, a) = \tau, \quad \text{if } a \in I$	TI2
$a \mathbin{\|\!\lfloor} x = a \cdot x$	CM2	$\chi(\tau_I, x \cdot y) = \chi(\tau_I, x) \cdot \chi(\tau_I, y)$	TI3
$(a \cdot x) \mathbin{\|\!\lfloor} y = a \cdot (x \parallel y)$	CM3		
$(x + y) \mathbin{\|\!\lfloor} z = x \mathbin{\|\!\lfloor} z + y \mathbin{\|\!\lfloor} z$	CM4	$\chi(f \circ g, x) = \chi\big(f, \chi(g, x)\big)$	XC1
$(a \cdot x) \mid b = (a \mid b) \cdot x$	CM5	$\chi\big((f \circ g) \circ h, x\big) = \chi\big(f \circ (g \circ h), x\big)$	XC2
$a \mid (b \cdot x) = (a \mid b) \cdot x$	CM6		
$(a \cdot x) \mid (b \cdot y) = (a \mid b) \cdot (x \parallel y)$	CM7	$\chi(f, \gamma) = \gamma$	X1
$(x + y) \mid z = x \mid z + y \mid z$	CM8	$\chi(f, \gamma \cdot x) = \gamma \cdot \chi(f, x)$	X2
$x \mid (y + z) = x \mid y + x \mid z$	CM9	$\chi(f, x + y) = \chi(f, x) + \chi(f, y)$	X3
$\chi(\partial_H, a) = a, \quad \text{if } a \notin H$	D1	$\chi(\pi_n, a) = a$	PR1
$\chi(\partial_H, a) = \delta, \quad \text{if } a \in H$	D2	$\chi(\pi_1, a \cdot x) = a$	PR2
$\chi(\partial_H, x \cdot y) = \chi(\partial_H, x) \cdot \chi(\partial_H, y)$	D3	$\chi(\pi_{n+1}, a \cdot x) = a \cdot \chi(\pi_n, x)$	PR3

Table 1. $ACP_{\tau,u}$.

Remark (2.3)

We will give a more explicit formulation of the definition of a linear functional specification. Let $N = \{n_1, \ldots, n_k\}$ be a set of function names. Let $\sigma : A \times \{1, \ldots, k\} \longrightarrow \{1, \ldots, k\}$ be a map. Now we define a linear functional specification to be the following:

$$E(N) = \big\{\, n_i(a) = a^{(i)} : a \in A,\ 1 \le i \le k \,\big\} \\ \cup \big\{\, n_i(a \cdot x) = a^{(i)} \cdot n_{\sigma(a,i)}(x) : a \in A,\ 1 \le i \le k \,\big\},$$

with $a^{(i)} \in A \cup \{\delta\}$. It will be clear that this formulation of a linear functional equation is equivalent to definition (2.2).

We will now formulate two principles which say something about the "solutions" for this sort of equational systems: the operator definition principle (*ODP*) and the operator specification principle (*OSP*). But first, we will need a definition to map the function names into the set of linear unary operators.

Definition (2.4)

We will call a mapping $\varphi : N \longrightarrow F$ a valuation for N.

Principle (2.5) *The Operator Definition Principle*

Let $E(N)$ be a linear functional specification for a set of function names N. Then the following holds: there is a valuation φ for N which solves the system of equations $E(N)$. We will use the compact notation *ODP* for this principle.

Principle (2.6) *The Operator Specification Principle*

Let $E(N)$ be a linear functional specification for a set of function names N. Then there is at most one valuation φ for N such that φ solves the system of equations $E(N)$. Here we will use the compact notation *OSP*.

Remarks (2.7)

See remarks (2.11) and (2.19) for examples of the use of both principles *ODP* and *OSP*. Henceforth, we will write for the expression $\chi(f, x)$, with $f \in F$ and $x \in P$, the more usual notation $f(x)$, provided that no confusion can arise. If n is a function name we will use this convention, too. If we have specified the boundary conditions of a certain function name n and we want this function name to respect the sequential composition, we will abbreviate this by stating the functional equation for n as follows: $n(a \cdot x) = n(a) \cdot n(x)$ instead of reiterating the results of $n(a)$ for each $a \in A$. Examples of linear functional specifications in which a function name distributes over the sequential composition can be found in remark (2.11) and in section 6.

Definition (2.8)

Let $f \in F$ be a linear unary operator. We will define a subset $D(f) \subseteq F$ as follows:

(i) $\quad f \in D(f)$

(ii) $\quad \big(g \in F,\, a \in A,\, b \in A \cup C,\, \forall x \in P : f(a \cdot x) = b \cdot g(x)\big) \Longrightarrow g \in D(f)$

(iii) $\quad h \in D(g),\, g \in D(f) \Longrightarrow h \in D(f).$

We will call $D(f)$ the set of derived operators of f.

Examples (2.9)

We will calculate here for a number of linear unary operators, their sets of derived operators. We will start with π_3. Because of condition (i), we see that $\pi_3 \in D(\pi_3)$. We know that for all $a \in A, x \in P : \pi_3(a \cdot x) = a \cdot \pi_2(x)$. So we find with (ii) that $\pi_2 \in D(\pi_3)$. We also know that $\pi_2(a \cdot x) = a \cdot \pi_1(x)$ for all $a \in A$ and $x \in P$; thus we find with (ii) that $\pi_1 \in D(\pi_2)$. And with the aid of (iii) we see that $\pi_1 \in D(\pi_3)$. Observe that we do not have *constant* linear unary operators. For, let $c : P \longrightarrow P$ be as follows. For all $x \in P : c(x) = c$, with $c \in P$. Suppose that $c \in F$, then we know that $c(\delta) = \delta$, because of X1. So we find that $c = \delta$. But on the other hand we know that $c(\tau) = \tau$, so we see that $c = \tau$. Since $\delta \neq \tau$ we see that the assumption that $c \in F$ cannot hold. So in particular, we do not have in F a linear unary operator ℓ, with for all $x \in P : \ell(x) = \tau$. We know that $\pi_1(a \cdot x) = a = a \cdot \tau = a \cdot \ell(x)$. But $\ell \notin F$. So we find that $D(\pi_1) = \{\pi_1\}$. We thus obtain $D(\pi_3) = \{\pi_1, \pi_2, \pi_3\}$. It is very easy to deduce that $D(\pi_n) = \{\pi_1, \ldots, \pi_n\}$, for $n \geq 1$. The facts that $D(\tau_I) = \{\tau_I\}$ and $D(\partial_H) = \{\partial_H\}$ are also easy to verify. In remark (2.19) we have two linear unary operators μ and ν. It is immediately clear that $D(\mu) = D(\nu) = \{\mu, \nu\}$ q.v..

Definition (2.10)

Let $f \in F$ be a linear unary operator, which can be defined with the aid of a linear functional specification. If $|D(f)| = 1$ we will call f a (linear unary) renaming operator.

Remark (2.11)

We will give the definition of the renaming operator as it is stated in [**17**], in order to make a comparison with the renaming operators that we defined hereinbefore in definition (2.10). Let $f : A \longrightarrow A \cup C$ be a function. Define an operator ρ_f as follows.

(*i*) $\rho_f(\gamma) = \gamma$

(*ii*) $\rho_f(a) = f(a)$

(*iii*) $\rho_f(x \cdot y) = \rho_f(x) \cdot \rho_f(y)$

(*iv*) $\rho_f(x + y) = \rho_f(x) + \rho_f(y)$

Consider the linear functional specification $E(r)$ for the set of function names $\{r\}$.

$$\begin{aligned} E(r) = \big\{\, & r(a) = f(a) : a \in A \,\big\} \\ & \cup \big\{\, r(a \cdot x) = r(a) \cdot r(x) : a \in A \,\big\}. \end{aligned}$$

According to *ODP* there is a valuation $\varphi : \{r\} \longrightarrow F$ which solves the system of equations $E(r)$. Let us say $\varphi(r) = \rho$. It is evident that $|D(\rho)| = 1$; so we know, by definition, that ρ is a renaming operator. We also see that ρ_f is a solution for this system: $\rho_f(a) = f(a)$ and $\rho_f(a \cdot x) = \rho_f(a) \cdot \rho_f(x)$. But according to *OSP*, there is at most one solution, so thus we find: $\rho_f = \rho$. We have seen that the "old" renaming operators can be defined in terms of the "new" ones. In fact, these definitions are equivalent, since a linear functional specification that defines a renaming operator can only be of the form that $E(r)$ has.

Definition (2.12)

A linear unary operator $f \in F$ is called an abstracting operator if there is a linear unary operator $g \in D(f)$, such that $g(a) = \tau$, for some $a \in A$. Otherwise, $f \in F$ is called a concrete operator. Observe that the abstraction operator $\tau_I \in F$ is an abstracting operator (if $I \neq \emptyset$).

Example (2.13)

Let μ, ν be the linear unary operators considered in (2.19). We already saw in (2.9) that μ is an element of $D(\nu)$. In the linear functional specification which defined both operators we can see that $\mu(i) = \tau$, so ν is an abstracting operator.

Subsequently, we will introduce two more principles: the recursive definition principle (*RDP*) and the recursive specification principle (*RSP*). For these principles, we need the notion of a (guarded) recursive specification. These notions are taken from [**7**], although the notion of a guard can already be found in [**9**]. The definitions of such specifications are given hereinafter.

Definition (2.14)

Let X be a set of variables. A recursive specification E with variable set X over $ACP_{\tau,u}$ is a system of recursion equations with variables in X:

$$E = \{x = t_x(X) : x \in X\}.$$

For all $x \in X$, we have that $t_x(X)$ is an $ACP_{\tau,u}$-term with variables from the set X.

Definition (2.15)

Let t be a term over $ACP_{\tau,u}$ without abstracting operators. Suppose that in t a variable x occurs. We will call an occurrence of x in t guarded if t has a subterm of the form $a \cdot s$, in which a is an atomic action and s is a term over $ACP_{\tau,u}$, which contains this occurrence of x. Otherwise we will call the occurrence of x in t unguarded. We will call an $ACP_{\tau,u}$-term t without abstracting operators guarded if all occurrences of all variables in t are guarded. Let $E = \{x = t_x(X) : x \in X\}$ be a recursive specification without abstracting operators. We will call E a guarded recursive specification if we can rewrite it to a recursive specification E' with the aid of the axioms and/or the aid of the specification E itself in which all right-hand sides of the recursion equations of E' are guarded. We will call E' a completely guarded recursive specification.

Definition (2.16)

Let $E = \{x = t_x(X) : x \in X\}$ be a recursive specification. A solution for E is a vector $p = (p_x)_{x \in X}$, with $p_x \in P$ for all $x \in X$, such that for all $x \in X$ the following expressions are true statements: $p_x = t_x(p)$, in which $t_x(p)$ is shorthand for: substitute for each occurrence of an element $x \in X$ in $t_x(X)$ the process p_x. We say that two solutions are equal if the components of the vectors are equal: $(p_x)_{x \in X} = (q_x)_{x \in X}$ if and only if we have for each $x \in X$ that $p_x = q_x$.

Principle (2.17) *The Recursive Definition Principle*

Let E be a guarded recursive specification. Then there is a solution for E.

Principle (2.18) *The Recursive Specification Principle*

Let E be a guarded recursive specification. Then there is at most one solution for E.

Remark (2.19)

Now we will explain why we have excluded all abstracting operators in definition (2.15). First, we will show that we have to exclude the abstraction operator itself. Consider the following recursive specification:

$$E = \left\{ x = i \cdot \tau_{\{j\}}(y),\, y = j \cdot \tau_{\{i\}}(x) \right\}.$$

Let there be an atomic action $a \in A \setminus \{i, j\}$, then it is easy to see that $(i \cdot a^n, j \cdot a^n)$ is a solution for E, for each $n \geq 1$. This means that we have to exclude the abstraction operator since *RSP* cannot be valid. The abstracting operators are of our next concern: with the aid of *ODP* it is very easy to make operators that behave like the abstraction operator. Consider the following linear functional specification for the set of names $N = \{n, m\}$.

$$\begin{aligned} E(N) = & \left\{ n(i) = j, n(j) = i, n(a) = a : a \in A \setminus \{i, j\} \right\} \\ & \cup \left\{ m(i) = \tau, m(j) = \tau, m(a) = a : a \in A \setminus \{i, j\} \right\} \\ & \cup \left\{ n(i \cdot x) = j \cdot m(x), n(j \cdot x) = i \cdot m(x), n(a \cdot x) = a \cdot n(x) : a \in A \setminus \{i, j\} \right\} \\ & \cup \left\{ m(i \cdot x) = \tau \cdot n(x), m(j \cdot x) = \tau \cdot n(x), m(a \cdot x) = a \cdot m(x) : a \in A \setminus \{i, j\} \right\}. \end{aligned}$$

According to *ODP*, there is a valuation $\varphi : N \longrightarrow F$, which solves the system $E(N)$. And by *OSP* there is at most one such a solution, so we can give these linear unary operators a name. Let us say, $\varphi(n) = \nu$ and $\varphi(m) = \mu$. Now consider the guarded recursive specification below:

$$E = \left\{ x = i \cdot \nu(y),\ y = j \cdot \nu(x) \right\}$$

Observe that we can prove the following for ν with induction to k:

$$\begin{aligned} \nu(a^k) &= a^k, \\ \nu(i^k) &= j^{\left[\frac{k+1}{2}\right]}, \\ \nu(j^k) &= i^{\left[\frac{k+1}{2}\right]}. \end{aligned}$$

Now it is easy to see that $(i^2 \cdot a^n, j^2 \cdot a^n)$ is a solution for this system E, for some $a \in A \setminus \{i, j\}$ and $n \in \mathbb{N}$. For:

$$\begin{aligned} i \cdot \nu(y) &= i \cdot \nu(j^2 \cdot a^n) \\ &= i^2 \cdot \mu(j \cdot a^n) \\ &= i^2 \cdot \tau \cdot \nu(a^n) \\ &= i^2 \cdot a^n \\ &= x. \end{aligned}$$

The calculation that $j \cdot \nu(x) = y$, is proved as above. This means that we found more than one solution for the guarded recursive specification above; hence, we have to exclude the abstracting operators, too, in the definition of a guarded recursive specification.

Observation (2.20)

In the above, we have just seen that if we want to use ACP_τ with renaming operators as they are defined in [**17**], we have to adjust the definition of a guarded recursive specification. For, we have to, at least, exclude all the renaming operators ρ, such that $\rho(a) = \tau$, for some atomic action $a \in A$ in the definition of a guarded recursive specification.

At this point we want to introduce the approximation induction principle. This definition can be found in [**8**].

Principle (2.21) *The Approximation Induction Principle*

Let x, y be $ACP_{\tau,u}$-terms. If we have for all $n \geq 1$ that $\pi_n(x) = \pi_n(y)$, then we have $x = y$. We will abbreviate this principle with *AIP*.

3. Termination

In this section we will prove the termination of a term rewriting system, associated with the axiom system $ACP_{\tau,u}$. We will exclude the axioms of commutativity of the alternative composition (A1) and the second and third τ-laws of Milner (T2–T3). First we will describe what exactly closed $ACP_{\tau,u}$-terms are.

Definition (3.1)

A closed term, or a closed $ACP_{\tau,u}$-term, is a term without variables of sort P and without variables of sort F.

Examples (3.2)

Suppose that we have just two atomic actions: $A = \{a, b\}$. We know that $\pi_1(a)$ is a closed term. We can rewrite this closed term to a term in which no elements of sort F occur. We see at once that $\pi_1(a)$ rewrites to a. Now let s be a function name. Consider the following linear functional specification.

$$E(s) = \{s(a) = b, s(b) = a\} \cup \{s(a \cdot x) = b \cdot s(x), s(b \cdot x) = a \cdot s(x)\}. \tag{1}$$

It is clear that $s(a^3)$ is a closed term. We can also rewrite this term with the aid of the linear functional specification $E(N)$. It is immediately clear that $s(a^3) = b^3$.

Remark (3.3)

We see that we can write the closed $ACP_{\tau,u}$-terms, considered above, without constants of sort F. This is what we want to prove. We will consider the term rewriting system associated with $ACP_{\tau,u}$ (see table 2) and we will also consider the equations of an arbitrary but fixed linear functional specification as rewriting rules from left to right. Thereinafter, we will study the termination of this whole system. The outline of this proof will be more or less the same as the proof of the termination of ACP_τ in appendix A of [**6**]. But we will use the lexicographical variant of the recursive path ordering to prove the termination. The lexicographical variant of the recursive path ordering can be found in [**13**]. A general reference to the subject of term rewriting is [**12**]. Another reference that can be useful is [**14**]. The following definition is taken from [**6**], with some additions specific to the present situation.

Definition (3.4)

Let x and y be terms, let $\alpha \in A \cup C$ and let $f \in F$. Then we define the *weight* of a closed term as follows:

(i)	$\lvert\alpha\rvert = 1,$	
(ii)	$\lvert x \star y\rvert = \lvert x\rvert + \lvert y\rvert$	for $\star = \cdot, \|, \mathbin{\|\!\lfloor}, \mid,$
(iii)	$\lvert x + y\rvert = \max\{\lvert x\rvert, \lvert y\rvert\},$	
(iv)	$\lvert\chi(f,x)\rvert = \lvert f(x)\rvert = \lvert x\rvert.$	

Definition (3.5) *The Partial Ordering of the Signature*

Because of problems with the reduction of the left-merge, the authors of [**6**] have introduced infinitely many operators: the so-called ranked operators. The rank of an operator $\|, \mathbin{\|\!\lfloor}, \mid$ is the weight of the subterm of which it is the leading operator. We will give the partial ordering of the operators. Let $m \geq 1$ and $n \geq 2$.

$$\|_n > \mathbin{\|\!\lfloor}_n, |_n \qquad \mathbin{\|\!\lfloor}_{n+1}, |_{n+1} > \|_n \qquad \chi, \|_n, \mathbin{\|\!\lfloor}_n, |_n > \cdot > + \qquad \pi_{m+1} > \pi_m.$$

Let N be a finite set of function names. The rank of a constant $n \in N$ is the weight of the subterm of which it is the leading operator. The set of these constants of rank k is denoted by N_k. The set of all these ranked constants of sort F is simply the union of all the N_k. We will denote this set by $N_r = \bigcup_{k=1}^{\infty} N_k$. We will give the partial ordering of these constants of sort F:

$$\forall\, n_k, m_l \in N_r : \quad k > l \Longrightarrow n_k > m_l.$$

For the remaining part of the signature we will define the partial ordering as follows. For all $n \in N_r$ and for all $a \in A$ we have: $n, |_2 > a > \tau > \delta$. See figure 2 for a graphical representation of this ordering.

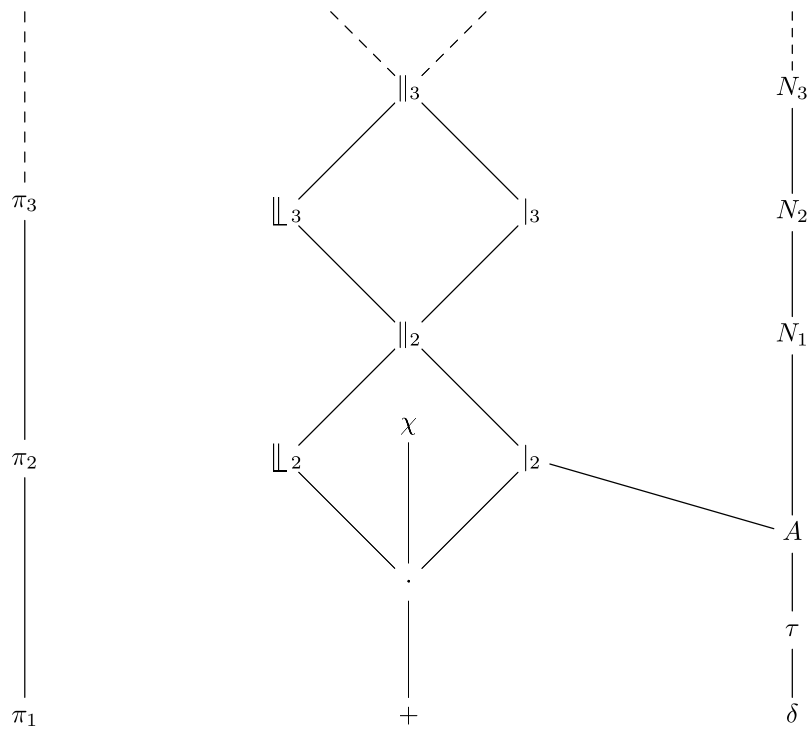

Figure 2. Visualization of the partial ordering.

Definition (3.6)

For each closed $ACP_{\tau,u}$-term t we obtain the the ranked term t_r by assigning to all operators their rank.

Example (3.7)

Here we will use the constant s that we defined in equation (1). Let

$$t = \big(s(a) \mathbin{⌊\!⌊} a \cdot b\big) \parallel \big(\tau \cdot s(a \cdot b) \mid s(a+b) \cdot a^2\big)$$

be a closed $ACP_{\tau,u}$-term. This term will be ranked as follows:

$$t_r = \big(s_1(a) \mathbin{⌊\!⌊}_3 a \cdot b\big) \parallel_{10} \big(\tau \cdot s_2(a \cdot b) \mid_7 \ s_2(a+b) \cdot a^2\big). \tag{2}$$

Definition (3.8)

Let D be the set of ranked closed $ACP_{\tau,u}$-terms. Let D^* be D where some of the symbols may be marked with an asterisk $(*)$. As an example let us take the ranked closed $ACP_{\tau,u}$-term t_r considered above in equation (2). Then we have that a typical element $t_r^* \in D^*$ is the following expression:

$$t_r^* = \big(s_1^*(a) \mathbin{⌊\!⌊}_3^* a \cdot b^*\big) \parallel_{10} \big(\tau^* \cdot s_2(a \cdot^* b) \mid_7 \ s_2(a^* + b) \cdot a^2\big).$$

Definition (3.9)

We will define a reduction relation "$\rightarrow$" on the set of marked ranked closed $ACP_{\tau,u}$-terms D^* as follows. For the sake of simplicity we will use, for the moment, prefix notation for the operators. Let H, G be function symbols occurring in the signature of $ACP_{\tau,u}$ (with this, we also mean constants such as atomic actions, δ, τ, or function names). Let $s, t, t_1, \ldots, t_k, s_1, \ldots, s_l$ be elements of D^*.

(i) $H(t_1, \ldots, t_k) \rightarrow H^*(t_1, \ldots, t_k), \quad (k \geq 0);$

(ii) $H^*(t_1, \ldots, t_k) \rightarrow G\big(H^*(t_1, \ldots, t_k), \ldots, H^*(t_1, \ldots, t_k)\big), \quad (H > G, \ k \geq 0);$

(iii) $H^*(t_1, \ldots, t_k) \rightarrow t_i, \quad (k \geq 1, \ 1 \leq i \leq k);$

(iv) $H^*\big(t_1, \ldots, G(s_1, \ldots, s_l), \ldots, t_k\big) \rightarrow H\big(t_1, \ldots, G^*(s_1, \ldots, s_l), \ldots, t_k\big), \quad (k \geq 1, \ l \geq 0);$

(v) $s \rightarrow t \Longrightarrow H(\ldots, s, \ldots) \rightarrow H(\ldots, t, \ldots);$

(vi) If $t \equiv H^*\big(G(s_1, \ldots, s_l), t_2, \ldots, t_k\big)$, then $t \rightarrow H\big(G^*(s_1, \ldots, s_l), t, \ldots, t\big), \quad (k \geq 1, \ l \geq 0).$

In which the ordering "$>$" on the signature of $ACP_{\tau,u}$ is already defined in definition (3.5). We will use the symbol "$\succ$" for the transitive closure of the above defined reduction relation.

Definition (3.10)

A partially ordered set $(S, >)$ consists of a set and a transitive and irreflexive binary relation $>$ defined on the elements of S. Notice that asymmetry of such a *strict* partial ordering follows from transitivity and irreflexivity. A partially ordered set $(S, >)$ is said to be *well-founded* if there are no infinite (strictly) descending sequences $s_1 > s_2 > s_3 > \cdots$ of elements of S.

Definition (3.11)

A term rewriting system over a set of terms has the termination property, if no infinite derivations are possible. A derivation is a sequence of rewrites.

It is known that the following theorem holds. See [**12**].

$(x + y) + z \to x + (y + z)$	RA2	$x \cdot \tau \to x$	RT
$x + x \to x$	RA3		
$(x + y) \cdot z \to x \cdot z + y \cdot z$	RA4	$\tau \mathbin{⌊\!⌊} x \to \tau \cdot x$	RTM1
$(x \cdot y) \cdot z \to x \cdot (y \cdot z)$	RA5	$(\tau \cdot x) \mathbin{⌊\!⌊} y \to \tau \cdot (x \parallel y)$	RTM2
$x + \delta \to x$	RA6	$\tau \mid x \to \delta$	RTC1
$\delta \cdot x \to \delta$	RA7	$x \mid \tau \to \delta$	RTC2
		$(\tau \cdot x) \mid y \to x \mid y$	RTC3
$a \mid b \to c_{a,b}$	RC	$x \mid (\tau \cdot y) \to x \mid y$	RTC4
$x \parallel y \to x \mathbin{⌊\!⌊} y + (y \mathbin{⌊\!⌊} x + x \mid y)$	RCM1	$\chi(\tau_I, a) \to a, \quad \text{if } a \notin I$	RTI1
$a \mathbin{⌊\!⌊} x \to a \cdot x$	RCM2	$\chi(\tau_I, a) \to \tau, \quad \text{if } a \in I$	RTI2
$(a \cdot x) \mathbin{⌊\!⌊} y \to a \cdot (x \parallel y)$	RCM3	$\chi(\tau_I, x \cdot y) \to \chi(\tau_I, x) \cdot \chi(\tau_I, y)$	RTI3
$(x + y) \mathbin{⌊\!⌊} z \to x \mathbin{⌊\!⌊} z + y \mathbin{⌊\!⌊} z$	RCM4		
$(a \cdot x) \mid b \to (a \mid b) \cdot x$	RCM5	$\chi(f \circ g, x) \to \chi\big(f, \chi(g, x)\big)$	RXC1
$a \mid (b \cdot x) \to (a \mid b) \cdot x$	RCM6	$\chi\big((f \circ g) \circ h, x\big) \to \chi\big(f \circ (g \circ h), x\big)$	RXC2
$(a \cdot x) \mid (b \cdot y) \to (a \mid b) \cdot (x \parallel y)$	RCM7	$\chi(f, \gamma) \to \gamma$	RX1
$(x + y) \mid z \to x \mid z + y \mid z$	RCM8	$\chi(f, \gamma \cdot x) \to \gamma \cdot \chi(f, x)$	RX2
$x \mid (y + z) \to x \mid y + x \mid z$	RCM9	$\chi(f, x + y) \to \chi(f, x) + \chi(f, y)$	RX3
$\chi(\partial_H, a) \to a, \quad \text{if } a \notin H$	RD1	$\chi(\pi_n, a) \to a$	RPR1
$\chi(\partial_H, a) \to \delta, \quad \text{if } a \in H$	RD2	$\chi(\pi_1, a \cdot x) \to a$	RPR2
$\chi(\partial_H, x \cdot y) \to \chi(\partial_H, x) \cdot \chi(\partial_H, y)$	RD3	$\chi(\pi_{n+1}, a \cdot x) \to a \cdot \chi(\pi_n, x)$	RPR3

Table 2. A term rewriting system associated with $ACP_{\tau,u}$.

Theorem (3.12) *Dershowitz*
Let (Σ, R) be a term rewriting system with finitely many rewriting rules and let "$>$" be a well-founded ordering on Σ. If $s \succ t$ for each rewriting rule $s \to t \in R$, then the term rewriting system (Σ, R) has the termination property. Where the arrow in $s \to t$ is, of course, not the arrow that we defined in definition (3.9), but ordinary notation for a rewriting rule.

At this point we are about to discuss the table of rewriting rules, associated with the axiom system $ACP_{\tau,u}$. See table 2 at page 13. In this table, we use the same notational conventions as in table 1. We have a rewriting rule "RC" instead of making rewriting rules of the axioms C1–C3 in table 1. In fact, we describe in these axioms some properties of the predefined communication function. We give in RC a "listing" of all the function applications, so we can rewrite the communication-merge in case we have a term containing an expression $a \mid b$, with $a, b \in A_\delta(= A \cup \delta)$. We have no rewriting rules that correspond with the axioms T2–T3. We have done this because these axioms do not have a clear direction: they can be used in both directions in order to simplify a certain term. Consider the following reduction:

$$\begin{aligned} \tau \cdot (a + b) + a &= \tau \cdot (a + b) + a + b + a \\ &= \tau \cdot (a + b) + a + b \\ &= \tau \cdot (a + b). \end{aligned}$$

In this example, we use twice the second τ-law of Milner; in both directions. With the third τ-law of Milner, the same difficulties arise.

Definition (3.13)

In remark (3.3) we already announced that we will add to the term rewriting system associated with $ACP_{\tau,u}$, a finite set of rewrite rules. Thence, let N be a set of function names for a linear functional specification $E(N)$. Recall that $E(N)$ is of the following form.

$$E(N) = \left\{ r_{n,a} \mid n \in N, a \in A \right\} \cup \left\{ e_{n,a} \mid n \in N, a \in A \right\}.$$

We will make this system of equations into a term rewriting system from the left to the right by simply substituting for an equality sign $=$ an arrow $\to$ in the boundary conditions and the functional equations. We will call the set of all these rules: the term rewriting system associated with the linear functional specification $E(N)$.

Theorem (3.14)

Let $E(N)$ be a linear functional specification. The rewriting rules in table 2 on page 13 together with the term rewriting system associated with the linear functional specification $E(N)$ have the termination property. See definition (3.11).

Proof. According to theorem (3.12), it is sufficient to prove, for each closed instance $s \to t$ of the rewriting rules that $s \succ t$. Let us first take a closer look at the rewriting rule RA2. We will make use of (vi).

$$\begin{aligned}(x+y)+z &\succ (x+y)+^* z\\ &\succ (x+^* y)+\big((x+y)+^* z\big)\\ &\succ x+\big((x+^* y)+z\big)\\ &\succ x+(y+z).\end{aligned}$$

Now let us treat RA3.

$$\begin{aligned}x+x &\succ x+^* x\\ &\succ x.\end{aligned}$$

This means that $x+x \succ x$. Now $\cdot > +$ so we find for RA4:

$$\begin{aligned}(x+y)\cdot z &\succ (x+y)\cdot^* z\\ &\succ (x+y)\cdot^* z+(x+y)\cdot^* z\\ &\succ (x+^* y)\cdot z+(x+^* y)\cdot z\\ &\succ x\cdot z+y\cdot z.\end{aligned}$$

Indeed, $(x+y)\cdot z \succ x\cdot z+y\cdot z$. Now we will treat RA5. We will use (vi), too. In fact, this case is proved analogously to RA2.

$$\begin{aligned}(x\cdot y)\cdot z &\succ (x\cdot y)\cdot^* z\\ &\succ (x\cdot^* y)\cdot\big((x\cdot y)\cdot^* z\big)\\ &\succ x\cdot\big((x\cdot^* y)\cdot z\big)\\ &\succ x\cdot(y\cdot z).\end{aligned}$$

So we see that $(x\cdot y)\cdot z \succ x\cdot(y\cdot z)$. The rewrite rules RA6 and RA7 are proved exactly the same as RA3. So let us verify RC. We make use of the fact that the communication-merge of rank two is greater than all atomic actions. Observe that we also have $|_2 > \delta$. We are to show that $a \mid_2 b \succ c_{a,b}$, with $c_{a,b} \in A_\delta$.

$$\begin{aligned} a \mid_2 b &\succ a \mid_2^* b \\ &\succ c_{a,b}. \end{aligned}$$

Now let us take a closer look at the merge. First we will handle RCM1. Let $|x| + |y| = n$. Notice that we are to show:

$$x \parallel_n y \succ x \mathbin{⫦}_n y + (y \mathbin{⫦}_n x + x \mid_n y).$$

We will make use of the fact that $\parallel_n > +$ and that $\parallel_n > \mathbin{⫦}_n, |_n$.

$$\begin{aligned} x \parallel_n y &\succ x \parallel_n^* y \\ &\succ x \parallel_n^* y + x \parallel_n^* y \\ &\succ x \parallel_n^* y + (x \parallel_n^* y + x \parallel_n^* y) \\ &\succ (x \parallel_n^* y) \mathbin{⫦}_n (x \parallel_n^* y) + \big((x \parallel_n^* y) \mathbin{⫦}_n (x \parallel_n^* y) + (x \parallel_n^* y) \mid_n (x \parallel_n^* y)\big) \\ &\succ x \mathbin{⫦}_n y + (y \mathbin{⫦}_n x + x \mid_n y). \end{aligned}$$

We will verify RCM2. Let $|x| = n$.

$$\begin{aligned} a \mathbin{⫦}_{n+1} x &\succ a \mathbin{⫦}_{n+1}^* x \\ &\succ \big(a \mathbin{⫦}_{n+1}^* x\big) \cdot \big(a \mathbin{⫦}_{n+1}^* x\big) \\ &\succ a \cdot x. \end{aligned}$$

Now we will handle the case RCM3. We will make use of the ranking of the operators. Let $|x|+|y| = n$. Then we easily find:

$$\begin{aligned} (a \cdot x) \mathbin{⫦}_{n+1} y &\succ (a \cdot x) \mathbin{⫦}_{n+1}^* y \\ &\succ \big((a \cdot x) \mathbin{⫦}_{n+1}^* y\big) \cdot \big((a \cdot x) \mathbin{⫦}_{n+1}^* y\big) \\ &\succ (a \cdot x) \cdot \Big(\big((a \cdot x) \mathbin{⫦}_{n+1}^* y\big) \parallel_n \big((a \cdot x) \mathbin{⫦}_{n+1}^* y\big)\Big) \\ &\succ (a \cdot^* x) \cdot \big((a \cdot x) \parallel_n y\big) \\ &\succ a \cdot \big((a \cdot^* x) \parallel_n y\big) \\ &\succ a \cdot (x \parallel_n y). \end{aligned}$$

We will consider RCM4. Let $|x| + |y| + |z| = p$, $|x| + |z| = q$ and $|y| + |z| = r$. We want to deduce that:

$$(x + y) \mathbin{⫦}_p z \succ x \mathbin{⫦}_q z + y \mathbin{⫦}_r z.$$

Now contemplate the following calculation.

$$\begin{aligned} (x + y) \mathbin{⫦}_p z &\succ (x + y) \mathbin{⫦}_p^* z \\ &\succ \big((x + y) \mathbin{⫦}_p^* z\big) + \big((x + y) \mathbin{⫦}_p^* z\big) \\ &\succ \big((x +^* y) \mathbin{⫦}_p z\big) + \big((x +^* y) \mathbin{⫦}_p z\big) \\ &\succ x \mathbin{⫦}_p z + y \mathbin{⫦}_p z \\ &\succ x \mathbin{⫦}_p^* z + y \mathbin{⫦}_p^* z \\ &\succ \big((x \mathbin{⫦}_p^* z) \mathbin{⫦}_q (x \mathbin{⫦}_p^* z)\big) + \big((y \mathbin{⫦}_p^* z) \mathbin{⫦}_r (y \mathbin{⫦}_p^* z)\big) \\ &\succ x \mathbin{⫦}_q z + y \mathbin{⫦}_r z. \end{aligned}$$

Indeed, we find that $(x+y) \mathbin{\|\!\_}_p z \succ x \mathbin{\|\!\_}_q z + y \mathbin{\|\!\_}_r z$. Let us take a look at RCM5. Let $|x| = n$. Then we are to show: $(a \cdot x) \mid_{n+2} b \succ (a \mid_2 b) \cdot x$. Consider thereunto the display below:

$$\begin{aligned}
(a \cdot x) \mid_{n+2} b &\succ (a \cdot x) \mid^*_{n+2} b \\
&\succ \big((a \cdot x) \mid^*_{n+2} b\big) \cdot \big((a \cdot x) \mid^*_{n+2} b\big) \\
&\succ \big((a \cdot^* x) \mid_{n+2} b\big) \cdot (a \cdot x) \\
&\succ (a \mid_{n+2} b) \cdot (a \cdot^* x) \\
&\succ (a \mid^*_{n+2} b) \cdot x \\
&\succ \big((a \mid^*_{n+2} b) \mid_2 (a \mid^*_{n+2} b)\big) \cdot x \\
&\succ (a \mid_2 b) \cdot x.
\end{aligned}$$

The deduction for the rewriting rule RCM6 is the same as the deduction above. So let us verify RCM7. Let $|x| + |y| = n$.

$$\begin{aligned}
(a \cdot x) \mid_{n+2} (b \cdot y) &\succ (a \cdot x) \mid^*_{n+2} (b \cdot y) \\
&\succ \big((a \cdot x) \mid^*_{n+2} (b \cdot y)\big) \cdot \big((a \cdot x) \mid^*_{n+2} (b \cdot y)\big) \\
&\succ \Big(\big((a \cdot x) \mid^*_{n+2} (b \cdot y)\big) \mid_2 \big((a \cdot x) \mid^*_{n+2} (b \cdot y)\big)\Big) \\
&\qquad \cdot \Big(\big((a \cdot x) \mid^*_{n+2} (b \cdot y)\big) \parallel_n \big((a \cdot x) \mid^*_{n+2} (b \cdot y)\big)\Big) \\
&\succ \big((a \cdot x) \mid_2 (b \cdot y)\big) \cdot \big((a \cdot x) \parallel_n (b \cdot y)\big) \\
&\succ \big((a \cdot^* x) \mid_2 (b \cdot^* y)\big) \cdot \big((a \cdot^* x) \parallel_n (b \cdot^* y)\big) \\
&\succ (a \mid_2 b) \cdot (x \parallel_n y).
\end{aligned}$$

We will treat RCM8. Let $p = |x| + |y| + |z|$, $q = |x| + |z|$ and $r = |y| + |z|$. Observe that we must show that: $(x+y) \mid_p z \succ x \mid_q z + y \mid_r z$. Consider the following:

$$\begin{aligned}
(x+y) \mid_p z &\succ (x+y) \mid^*_p z \\
&\succ \big((x+y) \mid^*_p z\big) + \big((x+y) \mid^*_p z\big) \\
&\succ \big((x +^* y) \mid_p z\big) + \big((x +^* y) \mid_p z\big) \\
&\succ x \mid_p z + y \mid_p z \\
&\succ x \mid^*_p z + y \mid^*_p z \\
&\succ \big((x \mid^*_p z) \mid_q (x \mid^*_p z)\big) + \big((y \mid^*_p z) \mid_r (y \mid^*_p z)\big) \\
&\succ x \mid_q z + y \mid_r z.
\end{aligned}$$

RCM9 is treated analogously. Let us calculate RD1 and RD2. This can be done in one calculation. We will use the fact that for all atomic actions $a \in A : a > \delta$.

$$\begin{aligned}
\chi(\partial_H, a) &\succ \chi^*(\partial_H, a) \\
&\succ a \\
&\succ a^* \\
&\succ \delta.
\end{aligned}$$

We see that $\chi(\partial_H, a) \succ a \succ \delta$, so with this, we handled both RD1 and RD2. We will treat RD3.

$$\begin{aligned}
\chi(\partial_H, x \cdot y) &\succ \chi^*(\partial_H, x \cdot y) \\
&\succ \chi^*(\partial_H, x \cdot y) \cdot \chi^*(\partial_H, x \cdot y) \\
&\succ \chi(\partial_H, x \cdot^* y) \cdot \chi(\partial_H, x \cdot^* y) \\
&\succ \chi(\partial_H, x) \cdot \chi(\partial_H, y).
\end{aligned}$$

RT is proved just like, e.g., RA7. RTM1 goes like RCM2. RTM2 is as RCM3. Now we will show RTC1. We will use that $\tau > \delta$. Let $|x| = n$.

$$\begin{aligned}\tau \mid_{n+1} x &\succ \tau \mid^*_{n+1} x\\ &\succ \tau\\ &\succ \tau^*\\ &\succ \delta.\end{aligned}$$

Of course RTC2 goes the same. We will show RTC3. Let $n = |x| + |y|$.

$$\begin{aligned}(\tau \cdot x) \mid_{n+1} y &\succ (\tau \cdot x) \mid^*_{n+1} y\\ &\succ (\tau \cdot^* x) \mid_{n+1} y\\ &\succ x \mid_{n+1} y\\ &\succ x \mid^*_{n+1} y\\ &\succ \bigl(x \mid^*_{n+1} y\bigr) \mid_n \bigl(x \mid^*_{n+1} y\bigr)\\ &\succ x \mid_n y.\end{aligned}$$

So we find $(\tau \cdot x) \mid_{n+1} y \succ x \mid_n y$. The proof of RTC4 is the same. RTI1–3 are proved in the same way as RD1–3. Observe that we use that for all atomic actions $a \in A : a > \tau$ in RTI1–2. Let us take RXC1. In this deduction we will make use of (vi) for the function symbol χ.

$$\begin{aligned}\chi(f \circ g, x) &\succ \chi^*(f \circ g, x)\\ &\succ \chi\bigl(f \circ^* g, \chi^*(f \circ g, x)\bigr)\\ &\succ \chi\bigl(f, \chi(f \circ^* g, x)\bigr)\\ &\succ \chi\bigl(f, \chi(g, x)\bigr).\end{aligned}$$

To deduce the desired inequality for RXC2, we make use of (vi) for the function symbol $\circ$.

$$\begin{aligned}\chi\bigl((f \circ g) \circ h, x\bigr) &\succ \chi^*\bigl((f \circ g) \circ h, x\bigr)\\ &\succ \chi\bigl((f \circ g) \circ^* h, x\bigr)\\ &\succ \chi\bigl((f \circ^* g) \circ \bigl((f \circ g) \circ^* h\bigr), x\bigr)\\ &\succ \chi\bigl(f \circ \bigl((f \circ^* g) \circ h\bigr), x\bigr)\\ &\succ \chi\bigl(f \circ (g \circ h), x\bigr).\end{aligned}$$

The case RX1 is trivial. So let us treat RX2.

$$\begin{aligned}\chi(f, \gamma \cdot x) &\succ \chi^*(f, \gamma \cdot x)\\ &\succ \chi^*(f, \gamma \cdot x) \cdot \chi^*(f, \gamma \cdot x)\\ &\succ (\gamma \cdot x) \cdot \chi(f, \gamma \cdot^* x)\\ &\succ (\gamma \cdot^* x) \cdot \chi(f, x)\\ &\succ \gamma \cdot \chi(f, x).\end{aligned}$$

For RX3 we will make use of the fact that $\chi > +$.

$$\begin{aligned}\chi(f, x + y) &\succ \chi^*(f, x + y)\\ &\succ \chi^*(f, x + y) + \chi^*(f, x + y)\\ &\succ \chi(f, x +^* y) + \chi(f, x +^* y)\\ &\succ \chi(f, x) + \chi(f, y).\end{aligned}$$

RPR1 goes like, e.g., RD1. Let us verify RPR2.

$$\begin{aligned}
\chi(\pi_1, a \cdot x) &\succ \chi^*(\pi_1, a \cdot x) \\
&\succ a \cdot x \\
&\succ a \cdot^* x \\
&\succ a.
\end{aligned}$$

For the verification of RPR3 we will use that $\pi_{n+1} > \pi_n$.

$$\begin{aligned}
\chi(\pi_{n+1}, a \cdot x) &\succ \chi^*(\pi_{n+1}, a \cdot x) \\
&\succ \chi^*(\pi_{n+1}, a \cdot x) \cdot \chi^*(\pi_{n+1}, a \cdot x) \\
&\succ (a \cdot x) \cdot \chi(\pi^*_{n+1}, a \cdot x) \\
&\succ (a \cdot^* x) \cdot \chi(\pi_n, a \cdot x) \\
&\succ a \cdot \chi^*(\pi_n, a \cdot x) \\
&\succ a \cdot \chi(\pi_n, x)
\end{aligned}$$

Let us now take a boundary condition. Recall that we have introduced the rank of a function name, thus for the boundary condition we are to show $\chi(n_1, a) \succ b$. We will make use of the fact that $n_1 > b$.

$$\begin{aligned}
\chi(n_1, a) &\succ \chi^*(n_1, a) \\
&\succ n_1 \\
&\succ n_1^* \\
&\succ b.
\end{aligned}$$

Indeed, we see that $\chi(n_1, a) \succ b$. Let us take a functional equation. Let $|x| = p$. We will show that $\chi(n_{p+1}, a \cdot x) \succ b \cdot \chi(m_p, x)$.

$$\begin{aligned}
\chi(n_{p+1}, a \cdot x) &\succ \chi^*(n_{p+1}, a \cdot x) \\
&\succ \chi^*(n_{p+1}, a \cdot x) \cdot \chi^*(n_{p+1}, a \cdot x) \\
&\succ n_{p+1} \cdot \chi(n_{p+1}, a \cdot^* x) \\
&\succ n^*_{p+1} \cdot \chi(n_{p+1}, x) \\
&\succ b \cdot \chi^*(n_{p+1}, x) \\
&\succ b \cdot \chi(n^*_{p+1}, x) \\
&\succ b \cdot \chi(m_p, x).
\end{aligned}$$

And finally, we find $\chi(n_{p+1}, a \cdot x) \succ b \cdot \chi(m_p, x)$. This ends the proof of (3.14).

Remarks (3.15)

The partial ordering on the signature in [**6**] differs in two ways from the partial ordering that is given in (3.5), or equivalently in figure 2. Firstly, we give the following ordering on atomic actions: for all $a \in A$, we defined $a > \tau > \delta$. In [**6**], there is no ordering on the atomic actions, nor on δ or τ, whatsoever. The author did not succeed in proving that $\partial_H(a) \succ \delta$, $\tau \mid x \succ \delta$, $x \mid \tau \succ \delta$ or $\tau_I(a) \succ \tau$, without some kind of ordering on the atomic actions. Secondly, in [**6**] there is no rewriting rule for $a \mid b$, with $a, b \in A \setminus \{\delta\}$. This is necessary in order to be able to prove an elimination result. This is also stated in [**6**], but without a rewriting rule for $a \mid b$, this result in [**6**] is not entirely correct.

We have seen that this way of proving termination is highly usable in process algebra. For, the method presented in [**6**], is generalized effortlessly to the present situation. Therefore, it is worthwhile investigating this method separately. However, we will not do that in this paper. For another way of proving termination in process algebra we refer to [**1**].

Here we will discuss here an elimination theorem which states that we can eliminate in a closed $ACP_{\tau,u}$-term all the operators that are not the alternative composition or the sequential composition. In other words, we can rewrite each closed $ACP_{\tau,u}$-term to a $BPA_{\delta,\tau}$-term. The acronym *BPA* stands for basic process algebra. This system consists of the first five laws of *PA*, which has been studied in [**7**]. The abbreviation *PA* stands for process algebra. The subscripts δ and τ, mean that the axioms concerning those special constants are added to the theory *BPA*. A good general reference to *BPA*, $BPA_{\delta,\tau}$ and *PA* is [**5**].

We will use the same notational conventions as before in table 1. For the axioms of $BPA_{\delta,\tau}$, see table 3 on page 19.

$x + y = y + x$	A1	$x \cdot \tau = x$	T1
$x + (y + z) = (x + y) + z$	A2	$\tau \cdot x + x = \tau \cdot x$	T2
$x + x = x$	A3	$a \cdot (\tau \cdot x + y) = a \cdot (\tau \cdot x + y) + a \cdot x$	T3
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4		
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5		
$x + \delta = x$	A6		
$\delta \cdot x = \delta$	A7		

Table 3. $BPA_{\delta,\tau}$.

Lemma (3.16)

Let t be a closed $BPA_{\delta,\tau}$-term. Let RA3–7 and RT from table 2 on page 13 be the term rewriting system associated with $BPA_{\delta,\tau}$, then we can rewrite the term t in one of the following forms:

$$u = \begin{cases} a, & \text{with } a \in A; \\ \delta; & \\ \tau; & \\ a \cdot v, & \text{with } a \in A_\delta \text{ and } v \text{ a closed } BPA_{\delta,\tau}\text{-term in normal form;} \\ v + w, & \text{with } v, w \text{ closed } BPA_{\delta,\tau}\text{-terms in normal form.} \end{cases}$$

Or, equivalently, there are closed $BPA_{\delta,\tau}$-terms $x_1, \ldots, x_n, t_1, \ldots, t_p$ and $n, p \geq 0$ such that

$$u = \sum_{i=1}^{n} a_i \cdot x_i + \sum_{k=1}^{p} \tau \cdot t_k,$$

for certain atomic actions $a_1, \ldots, a_n \in A$.

Proof. This is a well-known fact. See, e.g., [**5**]. Note that we rewrite modulo A1 and A2. We did not do that in theorem (3.14); we excluded the axiom A1 among others in there in order to be able to use the method of the recursive path ordering.

Theorem (3.17) *The Elimination Theorem*

Let t be a closed $ACP_{\tau,u}$-term. Then there is a closed $BPA_{\delta,\tau}$-term s, such that

$$ACP_{\tau,u} \vdash t = s.$$

Proof. Let N be the set of function names that occur in the closed $ACP_{\tau,u}$-term t. Rewrite this term t with the aid of the term rewriting system in table 2 on page 13 and the term rewriting system associated with $E(N)$* to a normal form. With the aid of theorem (3.14), we know that this is not an infinite process, i.e., there is a finite row:

$$t = t_0 \to t_1 \to \cdots \to t_n = s, \tag{3}$$

and we cannot perform any rewriting rule on s. We see immediately that in s we have not a merge operator, for otherwise we could apply RCM1. We will distinguish five cases. If in s occurs a function name, a left-merge, a communication-merge, an encapsulation operator or an abstraction operator, we will take a minimal subterm (in the sense of a minimal number of symbols), in which precisely one of these operators occurs.

1 We have a minimal subterm of the form $n(u)$, with $n \in N$ and u a closed $BPA_{\delta,\tau}$-term. Then we know that u has one of the forms displayed in (3.16). If $u = a$, then we can use a boundary condition: $\chi(n, a) \to b$, but this is in contradiction with the assumption that s is in normal form. If $u = \delta$ or τ, we can apply RX1. If $u = a \cdot v$, we can use a functional equation: $\chi(n, a \cdot v) \to b \cdot \chi(m, v)$. And if $u = v + w$, we can apply RX3.

2 We have a minimal subterm of the form $u_1 \mathbin{⌊\!⌊} u_2$, with u_1, u_2 closed $BPA_{\delta,\tau}$-terms. If $u_1 = a \in A_\delta$, then we can apply RCM2. If $u_1 = \tau$, we can use RTM1. If $u_1 = a \cdot v$, we can use RCM3. If $u_1 = v + w$, then we can use RCM4.

3 We have a minimal subterm of the form $u_1 \mid u_2$ with u_1, u_2 closed $BPA_{\delta,\tau}$-terms. If $u_1 = a \in A$ or δ, then we may use RC or RCM6. If $u_1 = \tau$, we can apply RTC1. If $u_1 = a \cdot v$, we can use RCM5 or RCM7 or RCM9 or RTC2. If $u_1 = v + w$, we can use RCM8.

4 We have a minimal subterm of the form $\partial_H(u)$, with u a closed $BPA_{\delta,\tau}$-term. If $u = a \in A$ or δ, we can apply RD1 or RD2. If $u = \tau$, we can apply RDT. If $u = a \cdot v$, we can use RD4 and, if $u = v + w$, we can use RD3.

5 We have a minimal subterm of the form $\tau_I(u)$, with u a closed $BPA_{\delta,\tau}$-term. This case is treated exactly the same as the former case.

So in s a left-merge, a communication-merge, an encapsulation operator or an abstraction operator cannot occur. Thus, s is a closed $BPA_{\delta,\tau}$-term. If we replace the arrows by equality signs, in equation (3), we obtain a proof in $ACP_{\tau,u}$ of $t = s$. This will end the proof of (3.17).

Corollary (3.18)

Let t be a closed $ACP_{\tau,u}$-term. Then we can rewrite this term t into a closed term u which has the following form:

$$u = \sum_{i=1}^{n} a_i \cdot x_i + \sum_{k=1}^{p} \tau \cdot t_k,$$

for closed $BPA_{\delta,\tau}$-terms $x_1, \ldots, x_n, t_1, \ldots, t_p$, $n, p \geq 0$ and certain $a_1, \ldots, a_n \in A$.

* see definition (3.13)

4. Theorems

In several applications in which auxiliary linear unary operators have been used in the past, there was a need to deduce some basic properties on these operators in order to be able to prove the desired results. We can think of an application wherein the approximation induction principle is used to establish a result. See definition (2.21) for the formulation of *AIP*. Therefore, we often need to know whether or not this particular auxiliary operator commutes with the projection operators π_n. In this section we will prove that if f can be defined with the aid of a linear functional specification and, if f is a concrete operator, then for all processes x that contain no τs, we have

$$\pi_n \circ f(x) = f \circ \pi_n(x).$$

See theorem (4.8) for details. Moreover, this yields that if we work in the axiom system ACP_u (ACP_u is $ACP_{\tau,u}$ without abstraction; see table 5 on page 39, section 5), all the linear unary operators that can be defined with the aid of a linear functional specification, commute with the projection operators. For in there all the operators and processes are both concrete.

Another thing that we might want to know is which processes are fixed points of such an operator. We will prove two results on this. To formulate these theorems we will use the notion of stable atomic actions (with respect to this operator). We call an atomic action stable with respect to $f \in F$ if we have for all $x \in P$ that $f(a \cdot x) = a \cdot f(x)$, that is, if f meets such an atom, it will pass it and nothing will change: neither the atomic action, nor the operator itself.

It is intuitively clear that, for instance, $\rho \circ \rho = \rho$ for a renaming operator that only renames $a \in A$ into b. This can be proved within the framework of ACP_τ, for closed terms. But we actually feel that this must be valid for open terms. With the aid of *OSP*, we can prove this for open terms. In fact, we will prove some theorems on idempotent linear unary operators that can be defined with the aid of a linear functional specification.

It is not only the case that we might want to know some basic facts about auxiliary unary operators that we defined in the middle of a verification, but it can be the case that we want to know something concerning operators that we already know. For instance, it is immediately clear that $\tau_I \circ \tau_{\{i\}} = \tau_I$, for all $i \in I$. Again it is already possible to prove this for closed terms within ACP_τ. With the aid of *OSP* it is very trivial to show that this is true in general. We see that the operator $\tau_{\{i\}}$ is absorbed by τ_I. We will define the notions of a left and right-absorber. We will prove some theorems concerning this matter.

In [**3**] we can find conditional axioms. We will treat a theorem from which two of these axioms follow immediately. Just another thing that can be of interest, is the question if a linear unary operator commutes with the encapsulation operator, or with the abstraction operator. Or when do we have that ∂_H and τ_I commute? The latter question is also known as a conditional axiom. In fact, we will prove this conditional axiom as a corollary of theorem (4.24), in which we give some necessary conditions in a way that renaming operators commute. We extend this result to arbitrary linear unary operators that can be defined with the aid of a linear functional specification. In these theorems we will use the notion of stable atomic actions, too.

At last we will treat, for "historic" reasons, a theorem that gives the conditions such that

$$f \circ g(x \parallel y) = f(x) \parallel g(y).$$

This theorem can be found in [**2**]. We included it, since it can be seen as one of the first theorems on auxiliary linear unary operators. This theorem might look a bit strange at first sight, but when we take $f = g = \partial_H$, we find a very useful equation: $\partial_H(x \parallel y) = \partial_H(x) \parallel \partial_H(y)$, since ∂_H is idempotent. (Of course, we do have certain restrictions on x and y.)

Definition (4.1)

Let x be a closed $ACP_{\tau,u}$-term and let $a \in A$. The alphabet of a process x is the set of atomic actions that x can perform. We define inductively what the alphabet $\alpha(x)$ of this x is. If in x occurs a constant of sort F, we will rewrite this term x with corollary (3.18) to a term x' and we define $\alpha(x) := \alpha(x')$. Now let x be a closed $ACP_{\tau,u}$-term without elements of sort F.

(*i*) $\alpha(\delta) = \alpha(\tau) = \emptyset$

(*ii*) $\alpha(a) = \{a\}$

(*iii*) $\alpha(\delta \cdot x) = \emptyset$

(*iv*) $\alpha(\tau \cdot x) = \alpha(x)$

(*v*) $\alpha(a \cdot x) = \{a\} \cup \alpha(x)$

(*vi*) $\alpha(x + y) = \alpha(x) \cup \alpha(y)$

Now we have defined the alphabet for closed terms, we will define it for terms t, with the property that $\pi_n(t)$ is a closed $ACP_{\tau,u}$-term.

(*vii*) $\alpha(t) = \bigcup_{n=1}^{\infty} \alpha\big(\pi_n(t)\big)$

This definition is taken from [**3**].

Definition (4.2)

Let $f \in F$ be a unary operator. An atomic action $a \in A$ is called *stable* (with respect to f), if we have for all $x \in P$:

$$f(a \cdot x) = a \cdot f(x).$$

Otherwise, it is called *unstable*. The set of all stable atomic actions with respect to f is denoted:

$$S(f) = \big\{\, a \in A \;\big|\; \forall x \in P : f(a \cdot x) = a \cdot f(x) \,\big\}.$$

We will use the notation $U(f) = A \setminus S(f)$ for the set of unstable atomic actions with respect to f.

Examples (4.3)

Let $I \subseteq A$. Then the set of unstable atomic actions of the abstraction operator is $U(\tau_I) = I$. Let $H \subseteq A$. Then the set of unstable atomic actions of the encapsulation operator is $U(\partial_H) = H$. Let $n \geq 1$. Then the set of stable atomic actions of the projection operator is $S(\pi_n) = \emptyset$.

Remark (4.4)

Let $f \in F$ be a linear unary operator. Let $a \in A$ be a stable atom with respect to f. Then we find that $f(a) = a$. For consider the following:

$$\begin{aligned} f(a) &= f(a \cdot \tau) \\ &= a \cdot f(\tau) \\ &= a \cdot \tau \\ &= a. \end{aligned}$$

Definition (4.5)

A process $x \in P$ is called concrete, if it has the following form:

$$x = \sum_{i=1}^{n} a_i \cdot x_i + \sum_{j=1}^{m} b_j,$$

for $a_1, \ldots, a_n, b_1, \ldots, b_m \in A_\delta$ and $x_1, \ldots, x_n \in P$ have the same form as x, i.e., they can be denoted without any τ.

Theorem (4.6)

Let x be a concrete process, then for all $n \geq 1$, we have that $\pi_n(x)$ is a closed $ACP_{\tau,u}$-term.

Proof. We will prove (4.6) with induction to n. We know that x is of the following form:

$$x = \sum_{i=1}^{n} a_i \cdot x_i + \sum_{j=1}^{m} b_j,$$

and $x_1, \ldots, x_n$ are also concrete (by definition). Now let $n = 1$, then

$$\pi_1(x) = \sum_{i=1}^{n} a_i + \sum_{j=1}^{m} b_j,$$

and this is a closed term. Suppose that (4.6) is proved for n, then we prove it for $n + 1$. We know that for all $i = 1, \ldots, n : \pi_n(x_i) = t_i$ are closed terms, so we find for x:

$$\begin{aligned} \pi_{n+1}(x) &= \sum_{i=1}^{n} a_i \cdot \pi_n(x_i) + \sum_{j=1}^{m} b_j \\ &= \sum_{i=1}^{n} a_i \cdot t_i + \sum_{j=1}^{m} b_j, \end{aligned}$$

which is a closed $ACP_{\tau,u}$-term.

Remark (4.7)

Notice that we can define the alphabet of a concrete process with the aid of the former theorem, for we defined the alphabet inductively on closed $ACP_{\tau,u}$-terms and the projections of concrete processes are closed terms.

Theorem (4.8)

Let $f \in F$ be a concrete linear unary operator that can be defined with a linear functional specification (see definition (2.12) for the definition of a concrete operator). Let x be a concrete process. Then we have for all $n \geq 1$:

$$\pi_n \circ f(x) = f \circ \pi_n(x).$$

Proof. We will prove this theorem with induction to n. Recall that x has the following form:

$$x = \sum_{i=1}^{n} a_i \cdot x_i + \sum_{j=1}^{m} b_j,$$

for certain concrete $x_1, \ldots, x_n$. Let $n = 1$, then,

$$\begin{aligned} \pi_1 \circ f(x) &= \sum_{i=1}^{n} f(a_i) + \sum_{j=1}^{m} f(b_j) \\ &= \sum_{i=1}^{n} f\big(\pi_1(a_i \cdot x_i)\big) + \sum_{j=1}^{m} f\big(\pi_1(b_j)\big) \\ &= f \circ \pi_1(x). \end{aligned}$$

Suppose that (4.8) is valid for n, then we prove it for $n+1$. Observe that all $g \in D(f)$ are concrete, if f itself is concrete.

$$\begin{aligned}
\pi_{n+1} \circ f(x) &= \sum_{i=1}^{n} f(a_i) \cdot \pi_n \circ g_i(x_i) + \sum_{j=1}^{m} \pi_{n+1} \circ f(b_j) \\
&= \sum_{i=1}^{n} f(a_i) \cdot g_i \circ \pi_n(x_i) + \sum_{j=1}^{m} f(b_j) \\
&= \sum_{i=1}^{n} f\big(a_i \cdot \pi_n(x_i)\big) + \sum_{j=1}^{m} f\big(\pi_{n+1}(b_j)\big) \\
&= f \circ \pi_{n+1}(x).
\end{aligned}$$

Since $g_i \in D(f)$. This finishes the proof of theorem (4.8).

Lemma (4.9)

Let $f \in F$ be a linear unary operator, not necessarily definable by a linear functional specification, and let x be a closed $ACP_{\tau,u}$-term. Then we have the following:

$$\alpha(x) \subseteq S(f) \Longrightarrow f(x) = x.$$

Proof. We will prove (4.9) with induction to the number n of symbols of x. So let $n = 1$, then we have three possibilities for x: $x = a$, $x = \delta$, or $x = \tau$. Because of X1, the latter two cases are proved and, due to the fact that $\alpha(a) \subseteq S(f)$ and remark (4.4), we know that $f(a) = a$. Now let $n > 1$ and suppose that (4.9) holds for all closed terms with their number of symbols $< n$. Because of (3.18), we know that x has the following form:

$$x = \sum_{i=1}^{n} a_i \cdot x_i + \sum_{k=1}^{p} \tau \cdot t_k,$$

for closed $ACP_{\tau,u}$-terms $x_1, \ldots, x_n, t_1, \ldots, t_p$, $n, p \geq 0$ and atomic actions $a_1, \ldots, a_n \in A$. Now we see that $a_i \in \alpha(x) \subseteq S(f)$, hence, $f(a_i \cdot x_i) = a_i \cdot f(x_i)$, for $1 \leq i \leq n$. Consider the following:

$$\begin{aligned}
f(x) &= \sum_{i=1}^{n} f(a_i \cdot x_i) + \sum_{k=1}^{p} f(\tau \cdot t_k) \\
&= \sum_{i=1}^{n} a_i \cdot f(x_i) + \sum_{k=1}^{p} \tau \cdot f(t_k) \\
&= \sum_{i=1}^{n} a_i \cdot x_i + \sum_{k=1}^{p} \tau \cdot t_k \\
&= x.
\end{aligned}$$

Observe that we can use the induction hypothesis, since $\alpha(x_i), \alpha(t_k) \subseteq \alpha(x) \subseteq S(f)$. This ends the proof of (4.9).

Theorem (4.10)

Let $f \in F$ be a concrete linear unary operator that can be defined with a linear functional specification and let x be a concrete process, and suppose that $\alpha(x) \subseteq S(f)$, then x is a fixed point of f.

Proof. According to *AIP* it suffices to prove for all $n \geq 1$:

$$\pi_n \circ f(x) = \pi_n(x).$$

Let $n \geq 1$ be fixed. As both x and f are concrete, we know that $\pi_n \circ f(x) = f \circ \pi_n(x) = f\big(\pi_n(x)\big)$. With the aid of (4.6) and (4.9) we know that $f\big(\pi_n(x)\big) = \pi_n(x)$, since

$$\alpha\big(\pi_n(x)\big) \subseteq \bigcup_{i=1}^{\infty} \alpha\big(\pi_i(x)\big) = \alpha(x) \subseteq S(f).$$

Thus, we see that x is a fixed point of f.

Definition (4.11)

Let $f \in F$ be a linear unary operator. If we have $f^2 = f$, we will call f idempotent.

Theorem (4.12)

Let $f \in F$ be a renaming operator. Suppose that the following condition holds.

$$\forall\, \alpha \in f\big(U(f)\big) : f(\alpha) = \alpha,$$

then f is idempotent, that is, $f^2 = f$.

Proof. Let n be a function name and consider the following linear functional specification.

$$\begin{aligned} E(n) = \big\{\, & n(a) = f(a) : a \in A \,\big\} \\ & \cup \big\{\, n(a \cdot x) = n(a) \cdot n(x) : a \in A \,\big\}. \end{aligned}$$

We see immediately that f is a solution for this system. Now we will show that f^2 is also a solution for it. Let $a \in S(f)$, then we see that $f^2(a) = a = f(a)$. Let $a \in U(f)$, then we know that $f(a)$ is a fixed point of f. So we find again $f^2(a) = f(a)$. Since f is a renaming, we find $f^2(a{\cdot}x) = f^2(a){\cdot}f^2(x)$. We see that f^2 is a solution for the linear functional specification $E(n)$. But according to *OSP*, we know that there is at most one solution, so we find $f^2 = f$. This ends the proof of (4.12).

Theorem (4.13)

Let $f_1 \in F$ be a linear unary operator that can be defined with the aid of a linear functional specification. Let $D(f_1) = \{f_1, \ldots, f_k\}$ be the set of derived operators. Let

$$\sigma : A \times \{1, \ldots, k\} \longrightarrow \{1, \ldots, k\}$$

be defined as follows. If we have for all $x \in P : f_i(a \cdot x) = f_i(a) \cdot f_j(x)$, then we define $\sigma(a, i) = j$. Suppose that the following conditions hold,

(*i*) $\quad \forall\, \alpha \in f_i\big(U(f_i)\big) : f_i(\alpha) = \alpha$

(*ii*) $\quad \sigma(a, i) = \sigma\big(f_i(a), i\big)$

then f_i is idempotent, for all $1 \le i \le k$.

Proof. Let $N = \{n_1, \ldots, n_k\}$ be a set of function names. Consider the following linear functional specification.

$$\begin{aligned} E(N) = \big\{\, & n_i(a) = f_i(a) : a \in A,\ 1 \le i \le k \,\big\} \\ & \cup \big\{\, n_i(a \cdot x) = n_i(a) \cdot n_{\sigma(a,i)}(x) : a \in A,\ 1 \le i \le k \,\big\}. \end{aligned}$$

It will be clear that $f_1, \ldots, f_k$ is a solution for this system of equations. We will show that $f_1^2, \ldots, f_k^2$ is also a solution for this system. Choose an $i \in \{1, \ldots, k\}$. As in theorem (4.12), we see at once that $f_i^2(a) = f_i(a)$. We will handle the functional equations. Here, we will use the second condition.

$$\begin{aligned} f_i^2(a \cdot x) &= f_i\big(f_i(a) \cdot f_{\sigma(a,i)}(x)\big) \\ &= f_i^2(a) \cdot f_{\sigma(f_i(a),i)} \circ f_{\sigma(a,i)}(x) \\ &= f_i^2(a) \cdot f_{\sigma(a,i)}^2(x). \end{aligned}$$

With the aid of *OSP*, we see that $f_i^2 = f_i$, for all $1 \le i \le k$. This will end the proof of (4.13).

Definition (4.14)

Let $f, g \in F$. If $f \circ g = g$, we will call g a left-absorber for f; if $g \circ f = g$, we say that g is a right-absorber for f. If g is a left-absorber and a right-absorber for f, we just say that g is an absorber for f.

Theorem (4.15) *(Left-absorption)*

Let $f, g \in F$ be renaming operators. If we have

$$S(g) \cup g\big(U(g)\big) \subseteq S(f) \cup \{\delta, \tau\},$$

then g is a left-absorber for f.

Proof. We are to show that $f \circ g = g$. Let n be a function name. Consider the following linear functional specification.

$$E(n) = \big\{\, n(a) = g(a) : a \in A \,\big\} \cup \big\{\, n(a \cdot x) = n(a) \cdot n(x) : a \in A \,\big\}.$$

We see immediately that g is a solution for $E(n)$. It is very easy to deduce that $f \circ g$ is also a solution for it. Hence, with the aid of *OSP*, we find that $f \circ g = g$, so g is a left-absorber for f. This will end the proof of (4.15).

Theorem (4.16) *(Left-absorption)*

Let $f, g_1 \in F$. Suppose that f is a renaming operator and suppose that g_1 can be defined with the aid of a linear functional specification. Let $D(g_1) = \{g_1, \ldots, g_l\}$ be the set of derived operators of g_1. Suppose that the following condition holds:

$$\bigcap_{j=1}^{l} \Big(S(g_j) \cup g_j\big(U(g_j)\big) \Big) \subseteq S(f) \cup \{\delta, \tau\}.$$

Then all g_j are left-absorbers for f.

Proof. Let $N = \{n_1, \ldots, n_l\}$ be a set of function names. Define a map $\rho : A \times \{1, \ldots, l\} \longrightarrow \{1 \ldots, l\}$ as follows:

$$\rho(a, j) = k \iff \forall\, x \in P : g_j(a \cdot x) = g_j(a) \cdot g_k(x).$$

Consider the following linear functional specification.

$$\begin{aligned} E(N) = {} & \big\{\, n_j(a) = g_j(a) : a \in A,\ 1 \leq j \leq l \,\big\} \\ & \cup \big\{\, n_j(a \cdot x) = n_j(a) \cdot n_{\rho(a,j)}(x) : a \in A,\ 1 \leq j \leq l \,\big\}. \end{aligned}$$

It will be clear that $g_1, \ldots, g_l$ is a solution for this system of equations. We will show that

$$f \circ g_1, \ldots, f \circ g_l$$

is also a solution. It is very easy to see that $f \circ g_j(a) = g_j(a)$. Hence, all $f \circ g_j$ satisfy the boundary conditions of $E(N)$. We will show that the functional equations are satisfied, as well.

$$\begin{aligned} f \circ g_j(a \cdot x) &= f\big(g_j(a) \cdot g_{\rho(a,j)}(x)\big) \\ &= f \circ g_j(a) \cdot f \circ g_{\rho(a,j)}(x). \end{aligned}$$

We find thus, according to *OSP*, that $f \circ g_j = g_j$, and all g_j are left-absorbers for f. This ends the proof.

Observe that in these two absorption theorems, we could have replaced the conditions respectively by $f \circ g(a) = g(a)$ and $f \circ g_j(a) = g_j(a)$. We did *not* do that for orthogonality reasons: with the conditions as they are, we can formulate the following generalization.

Theorem (4.17) *(Left-absorption)*

Let $f_1, g_1 \in F$ be definable with the aid of linear functional specifications. Let their sets of derived operators be as follows.

$$D(f_1) = \{f_1, \ldots, f_k\},\ D(g_1) = \{g_1, \ldots, g_l\}.$$

Suppose that the following condition holds.

$$\bigcap_{j=1}^{l} \Big(S(g_j) \cup g_j\big(U(g_j)\big) \Big) \subseteq \bigcap_{i=1}^{k} S(f_i) \cup \{\delta, \tau\}.$$

Then all g_j are left-absorbers for all f_i.

Proof. We will use the notations of (4.16). We must show that $f_i \circ g_j$ is a solution for $E(N)$. First, we will handle the boundary conditions. Fix $1 \leq i \leq k$ and $1 \leq j \leq l$. First, let $a \in S(g_j)$, then we see that $a \in S(f_i)$, so $f_i \circ g_j(a) = a = g_j(a)$. Now suppose that $a \in U(g_j)$. If $g_j(a) = \gamma$, with $\gamma \in \{\delta, \tau\}$, then we have

$$\begin{aligned} f_i \circ g_j(a) &= f_i(\gamma) \\ &= \gamma \\ &= g_j(a). \end{aligned}$$

If $g_j(a) \in S(f_i)$, then we find immediately that $f_i \circ g_j(a) = g_j(a)$. Hence, the boundary conditions are treated. Now we will handle the functional equations. We will only treat the case that $a \in U(g_j)$ and $g_j(a) \neq \gamma$. So $g_j(a) \in S(f_i)$. We see that

$$\begin{aligned} f_i \circ g_j(a \cdot x) &= f_i\big(g_j(a) \cdot g_{\rho(a,j)}(x)\big) \\ &= f_i \circ g_j(a) \cdot f_i \circ g_{\rho(a,j)}(x). \end{aligned}$$

Hence, we obtain with the aid of *OSP* that $f_i \circ g_j = g_j$ for all i and j. This is what we wanted to prove.

Theorem (4.18) *(Right-absorption)*

Let $f, g \in F$ be renaming operators. Suppose that

$$\forall\, a \in A : f \circ g(a) = f(a),$$

then f is a right-absorber for g.

Proof. Let n be a function name. Consider the following linear functional specification.

$$E(n) = \big\{\, n(a) = f(a) : a \in A \,\big\} \cup \big\{\, n(a \cdot x) = n(a) \cdot n(x) : a \in A \,\big\}.$$

We see that f is a solution for $E(n)$. It is trivial to deduce that the same yields for $f \circ g$, so with *OSP*, we find that $f \circ g = f$. This is what we wanted to prove.

Theorem (4.19) *(Right-absorption)*

Let $f_1, g \in F$. Suppose that f_1 is definable with the aid of a linear functional specification, and let g be a renaming operator. Let $D(f_1) = \{f_1, \ldots, f_k\}$ be the set of derived operators of f_1. Define $\sigma : A \times \{1, \ldots, k\} \longrightarrow \{1, \ldots, k\}$ as follows

$$\sigma(a, i) = j \iff \forall\, x \in P : f_i(a \cdot x) = f_i(a) \cdot f_j(x).$$

Suppose that the following conditions are valid.

(i) $\quad f_i \circ g(a) = f_i(a),\ 1 \le i \le k,\ a \in A$

(ii) $\quad \sigma\big(g(a), i\big) = \sigma(a, i),\ 1 \le i \le k,\ a \in A$

Then all f_i are right-absorbers for g.

Proof. Let $N = \{n_1, \ldots, n_k\}$ be a set of function names. Consider the linear functional specification hereinafter.

$$\begin{aligned} E(N) = \big\{\, & n_i(a) = f_i(a) : a \in A, 1 \le i \le k \,\big\} \\ & \cup \big\{\, n_i(a \cdot x) = n_i(a) \cdot n_{\sigma(a,i)}(x) : a \in A, 1 \le i \le k \,\big\}. \end{aligned}$$

We see that $f_1, \ldots, f_k$ is a solution for this system of equations. We will show that this is also valid for $f_i \circ g$ with $1 \le i \le k$. Because of the first condition we will only have to treat the functional equations.

$$\begin{aligned} f_i \circ g(a \cdot x) &= f_i\big(g(a) \cdot g(x)\big) \\ &= f_i \circ g(a) \cdot f_{\sigma(g(a),i)} \circ g(x) \\ &= f_i \circ g(a) \cdot f_{\sigma(a,i)} \circ g(x). \end{aligned}$$

Observe that we used the second condition. We find thus, with the aid of *OSP* that $f_i \circ g = f_i$. This is precisely what we wanted to prove.

Theorem (4.19) will not generalize any further. For, suppose that $g \in F$ is also definable with the aid of a linear functional specification and suppose that $|D(g)| > 1$. Then there is an atomic action $a \in A$ and a linear unary operator $h \in D(g)$, such that for all $x \in P : g(a \cdot x) = g(a) \cdot h(x)$. But then we find with the second condition that $f_i \circ g(a \cdot x) = f_i \circ g(a) \cdot f_{\sigma(a,i)} \circ h(x)$, but this functional equation does not correspond with any of the functional equations in the linear functional specification $E(N)$ which defines $f_1, \ldots, f_k$. So we find that $f_i \circ g$ is not a solution for it. It turns out that the assumption that $|D(g)| > 1$ cannot hold. We find thus that g must be a renaming operator.

Theorem (4.20)

Let $f, f_1, \ldots, f_k$ be renaming operators. Suppose that the following holds:

$$\forall\, a \in A : f(a) = f_1 \circ f_2 \circ \cdots \circ f_k(a)$$

then $f = f_1 \circ f_2 \circ \cdots \circ f_k$.

Proof. Let n be a function name. Consider the following linear functional specification.

$$E(n) = \big\{\, n(a) = f(a) : a \in A \,\big\} \cup \big\{\, n(a \cdot x) = n(a) \cdot n(x) : a \in A \,\big\}.$$

It will be clear that f is a solution for this system. But we also see that $f_1 \circ f_2 \circ \cdots \circ f_k$ is a solution for it; so in accordance with *OSP* we may conclude that they are equal. This will end the proof of (4.20).

Corollary (4.21)

Let $H_1, H_2 \subseteq A$ and let $H = H_1 \cup H_2$. Then we have $\partial_H = \partial_{H_1} \circ \partial_{H_2}$.

Proof. It is trivial to verify that the conditions of theorem (4.20) are satisfied. With this, we conclude the proof of (4.21).

Corollary (4.22)

Let $I_1, I_2 \subseteq A$ and let $I = I_1 \cup I_2$. Then we have $\tau_I = \tau_{I_1} \circ \tau_{I_2}$.

Proof. Trivial.

Remarks (4.23)

Both corollaries (4.21) and (4.22) are known as conditional axioms. We can find these axioms among others in [**3**]. We see that it is very trivial to prove these statements, with this theory. But in the setting of ACP_τ it is only possible to prove these axioms for closed terms.

Theorem (4.24)

Let $f, g \in F$ be renaming operators. Suppose that the following holds:

(*i*) $S(f) \cup S(g) = A$,

(*ii*) $f\big(U(f)\big) \subseteq S(g) \cup \{\delta, \tau\}$,

(*iii*) $g\big(U(g)\big) \subseteq S(f) \cup \{\delta, \tau\}$,

then f and g commute, i.e., $f \circ g = g \circ f$.

Proof. Let n be a function name. Consider the linear functional specification $E(n)$ below:

$$\begin{aligned} E(n) = {} & \big\{\, n(a) = a : a \in S(f) \cap S(g) \,\big\} \\ & \cup \big\{\, n(a) = f(a) : a \in S(g) \setminus S(f) \,\big\} \\ & \cup \big\{\, n(a) = g(a) : a \in S(f) \setminus S(g) \,\big\} \\ & \cup \big\{\, n(a \cdot x) = n(a) \cdot n(x) : a \in A \,\big\}. \end{aligned}$$

First, we will show that $f \circ g$ is a solution for the linear functional specification above. If a is in $S(f) \cap S(g)$, then we see that $f \circ g(a) = a$. Now let $a \in S(g) \setminus S(f)$; then we see with (*ii*) that $f \circ g(a) = f(a)$. Let $a \in S(f) \setminus S(g)$; then we obtain with the aid of (*iii*) that $f \circ g(a) = g(a)$. Finally, we take $a \in A$ and $x \in P$; then we easily find that $f \circ g(a \cdot x) = f \circ g(a) \cdot f \circ g(x)$. This means that $f \circ g$ is a solution for the system $E(n)$. We can also show that $g \circ f$ is a solution for the linear functional specification above, so with the aid of *OSP*, we find that $f \circ g = g \circ f$. This ends the proof of (4.24).

Corollary (4.25)

Let $I, H \subseteq A$. Suppose that $I \cap H = \emptyset$. Then the encapsulation operator ∂_H and the abstraction operator τ_I commute.

Proof. We will verify the conditions of theorem (4.24). We know that $S(\partial_H) = A \setminus H$ and $S(\tau_I) = A \setminus I$, so because of the fact that $I \cap H = \emptyset$, we see immediately that (*i*) holds. We see that $\partial_H(H) = \{\delta\} \subseteq S(\tau_I) \cup \{\delta, \tau\}$, so (*ii*) is valid. The same applies to (*iii*), thus, we may use theorem (4.24) and we find that $\tau_I \circ \delta_H = \delta_H \circ \tau_I$. This ends the proof.

Remark (4.26)

Corollary (4.25) is also known as a conditional axiom. It can be found in [**3**]. In the setting of ACP_τ, it is possible to prove this for closed ACP_τ-terms, but in the framework of $ACP_{\tau,u}$, it is possible to prove this axiom for all processes.

Theorem (4.27)

Let $f \in F$ be a renaming operator. Let $k_1 \in F$ be a linear unary operator that can be defined with the aid of a linear functional specification. Let the set of derived operators of k_1 be $D(k_1) = \{k_1, \ldots, k_l\}$. Suppose that the following holds:

(i) $S(f) \cup \bigcap_{i=1}^{l} S(k_i) = A$

(ii) $f\big(U(f)\big) \subseteq \bigcap_{i=1}^{l} S(k_i) \cup \{\delta, \tau\}$

(iii) $\bigcup_{i=1}^{l} k_i\big(U(k_i)\big) \subseteq S(f) \cup \{\delta, \tau\}$

then f and k_i commute, i.e., for all i in $\{1, \ldots, l\}$, we have: $f \circ k_i = k_i \circ f$.

Proof. Define a map $\sigma : A \times \{1, \ldots, l\} \longrightarrow \{1, \ldots, l\}$ as follows. Let $a \in A$ and $1 \le i \le l$ be chosen. We know that there is an operator $k_j \in D(k_1)$, such that $k_i(a \cdot x) = b \cdot k_j(x)$, for a certain $a \in A \cup C$. Now we will define $\sigma(a, i) = j$. Let $M = \{m_1, \ldots, m_l\}$ be a set of function names. Consider the following linear functional specification. Let $C = \bigcap_{i=1}^{l} S(k_i)$.

$$\begin{aligned}
E(M) = &\big\{\, m_i(a) = a : a \in S(f) \cap C,\ 1 \le i \le l \,\big\} \\
&\cup \big\{\, m_i(a) = k_i(a) : a \in S(f) \setminus C,\ 1 \le i \le l \,\big\} \\
&\cup \big\{\, m_i(a) = f(a) : a \in C \setminus S(f),\ 1 \le i \le l \,\big\} \\
&\cup \big\{\, m_i(a \cdot x) = a \cdot m_i(x) : a \in S(f) \cap C,\ 1 \le i \le l \,\big\} \\
&\cup \big\{\, m_i(a \cdot x) = k_i(a) \cdot m_{\sigma(a,i)}(x) : a \in S(f) \setminus C,\ 1 \le i \le l \,\big\} \\
&\cup \big\{\, m_i(a \cdot x) = f(a) \cdot m_i(x) : a \in C \setminus S(f),\ 1 \le i \le l \,\big\}.
\end{aligned}$$

Let $x \in P$ and $i \in \{1, \ldots, l\}$ be fixed. Let $a \in S(f) \cap C$, then we see that $k_i \circ f(a) = a$ and we see that $f \circ k_i(a) = a$, too. Moreover, we see that

$$k_i \circ f(a \cdot x) = a \cdot k_i \circ f(x)$$

and

$$f \circ k_i(a \cdot x) = a \cdot f \circ k_i(x).$$

Now let $a \in S(f) \setminus C$. Then we also see that $k_i \circ f(a) = k_i(a)$, and we see that $f \circ k_i(a) = k_i(a)$, because of *(iii)*. Moreover, we see the following:

$$\begin{aligned}
f \circ k_i(a \cdot x) &= f\big(k_i(a) \cdot k_{\sigma(a,i)}(x)\big) \\
&= k_i(a) \cdot f \circ k_{\sigma(a,i)}(x)
\end{aligned}$$

and

$$\begin{aligned}
k_i \circ f(a \cdot x) &= k_i\big(a \cdot f(x)\big) \\
&= k_i(a) \cdot k_{\sigma(a,i)} \circ f(x).
\end{aligned}$$

Now let $a \in C \setminus S(f)$. Then we see that $k_i \circ f(a) = f(a)$ and we see that $f \circ k_i(a) = f(a)$. It is also easy to see that

$$\begin{aligned}
k_i \circ f(a \cdot x) &= k_i\big(f(a) \cdot f(x)\big) \\
&= f(a) \cdot k_i \circ f(x)
\end{aligned}$$

and

$$\begin{aligned}
f \circ k_i(a \cdot x) &= f\big(a \cdot k_i(x)\big) \\
&= f(a) \cdot f \circ k_i(x),
\end{aligned}$$

since f is a renaming operator. Thus, we find that $k_1 \circ f, \ldots, k_l \circ f$ is a solution for the linear functional specification above. But we also see that $f \circ k_1, \ldots, f \circ k_l$ is a solution for this system. So with the aid of *OSP*, we may conclude that $k_i \circ f = f \circ k_i$. Herewith we end the proof of (4.27).

Theorem (4.28)

Let $f_1, g_1 \in F$ be linear unary operators that can be defined with the aid of linear functional specifications. Let the sets of derived operators be given as follows:

$$D(f_1) = \{f_1, \ldots, f_n\},$$
$$D(g_1) = \{g_1, \ldots, g_m\}.$$

Suppose that the following holds:

(i) $\bigcap_{i=1}^{n} S(f_i) \cup \bigcap_{j=1}^{m} S(g_j) = A$

(ii) $\bigcup_{i=1}^{n} f_i\big(U(f_i)\big) \subseteq \bigcap_{j=1}^{m} S(g_j) \cup \{\delta, \tau\}$

(iii) $\bigcup_{j=1}^{m} g_i\big(U(g_i)\big) \subseteq \bigcap_{i=1}^{n} S(f_i) \cup \{\delta, \tau\}$

then all elements of the derived operator set $D(f_1)$ of f_1, commute with all elements of the derived operator set $D(g_1)$ of g_1.

Proof. Define two maps $\sigma : A \times \{1, \ldots, n\} \longrightarrow \{1, \ldots, n\}$ and $\rho : A \times \{1, \ldots, m\} \longrightarrow \{1, \ldots, m\}$ as follows. Let $a \in A$ and $1 \leq i \leq n$ be chosen. There is an $f_j \in D(f_1)$ such that $f_i(a \cdot x) = b \cdot f_j(x)$, for a certain $b \in A \cup C$. We define $\sigma(a, i) = j$. Now fix $c \in A$ and $1 \leq j \leq m$. We know that there is a $g_k \in D(g_1)$ such that $g_j(c \cdot x) = d \cdot g_k(x)$ for a certain $d \in A \cup C$. We define $\rho(c, j) = k$. Let $M = \{m_{i,j} : 1 \leq i \leq n, 1 \leq j \leq m\}$ be a set of function names. We will use the following abbreviations:

$$S_1 = \bigcap_{i=1}^{n} S(f_i) \text{ and } S_2 = \bigcap_{j=1}^{m} S(g_j).$$

Consider the linear functional specification hereinafter.

$$\begin{aligned}
E(M) = &\left\{ m_{i,j}(a) = a : a \in S_1 \cap S_2,\, 1 \leq i \leq n, 1 \leq j \leq m \right\} \\
&\cup \left\{ m_{i,j}(a) = g_j(a) : a \in S_1 \setminus S_2,\, 1 \leq i \leq n, 1 \leq j \leq m \right\} \\
&\cup \left\{ m_{i,j}(a) = f_i(a) : a \in S_2 \setminus S_1,\, 1 \leq i \leq n, 1 \leq j \leq m \right\} \\
&\cup \left\{ m_{i,j}(a \cdot x) = a \cdot m_{i,j}(x) : a \in S_1 \cap S_2,\, 1 \leq i \leq n, 1 \leq j \leq m \right\} \\
&\cup \left\{ m_{i,j}(a \cdot x) = g_j(a) \cdot m_{i,\rho(a,j)}(x) : a \in S_1 \setminus S_2,\, 1 \leq i \leq n, 1 \leq j \leq m \right\} \\
&\cup \left\{ m_{i,j}(a \cdot x) = f_i(a) \cdot m_{\sigma(a,i),j}(x) : a \in S_2 \setminus S_1,\, 1 \leq i \leq n, 1 \leq j \leq m \right\}.
\end{aligned}$$

Let $x \in P$, $1 \leq i \leq n$ and $1 \leq j \leq m$ be fixed. Choose $a \in S_1 \cap S_2$. We see immediately that $f_i \circ g_j(a) = a = g_j \circ f_i(a)$. It is also easy to see the following.

$$f_i \circ g_j(a \cdot x) = a \cdot f_i \circ g_j(x)$$

and

$$g_j \circ f_i(a \cdot x) = a \cdot g_j \circ f_i(x).$$

Now let $a \in S_1 \setminus S_2$. Then it is easy to see that $f_i \circ g_j(a) = g_j(a) = g_j \circ f_i(a)$. It is also immediately clear that

$$\begin{aligned}
f_i \circ g_j(a \cdot x) &= f_i\big(g_j(a) \cdot g_{\rho(a,j)}(x)\big) \\
&= g_j(a) \cdot f_i \circ g_{\rho(a,j)}(x)
\end{aligned}$$

and

$$\begin{aligned}
g_j \circ f_i(a \cdot x) &= g_j\big(a \cdot f_i(x)\big) \\
&= g_j(a) \cdot g_{\rho(a,j)} \circ f_i(x).
\end{aligned}$$

Observe that we used here (*iii*). Finally let $a \in S_2 \setminus S_1$. We see that $f_i \circ g_j(a) = f_i(a) = g_j \circ f_i(a)$. Moreover, using (*ii*), we find that

$$\begin{aligned} f_i \circ g_j(a \cdot x) &= f_i\big(a \cdot g_j(x)\big) \\ &= f_i(a) \cdot f_{\sigma(a,i)} \circ g_j(x) \end{aligned}$$

and

$$\begin{aligned} g_j \circ f_i(a \cdot x) &= g_j\big(f_i(a) \cdot f_{\sigma(a,i)}(x)\big) \\ &= f_i(a) \cdot g_j \circ f_{\sigma(a,i)}(x). \end{aligned}$$

Thus, we find that $\{f_i \circ g_j : 1 \leq i \leq n, 1 \leq j \leq m\}$ is a solution for $E(M)$. But we also find that $\{g_j \circ f_i : 1 \leq i \leq n, 1 \leq j \leq m\}$ is a solution for $E(M)$. So with the use of *OSP*, we find that $f_i \circ g_j = g_j \circ f_i$, for all $1 \leq i \leq n$ and $1 \leq j \leq m$. This ends the proof of (4.28).

Theorem (4.29)

Suppose that there is no communication, that is, $a \mid b = \delta$ for all atomic actions $a, b \in A$. Let $f_1, g_1 \in F$ be definable with the aid of linear functional specifications. Let their sets of derived operators be $D(f_1) = \{f_1, \ldots, f_n\}$ and $D(g_1) = \{g_1, \ldots, g_m\}$. Let x, y be closed $ACP_{\tau,u}$-terms. Suppose that the following conditions hold.

(*i*) $\alpha(x) \cup \bigcup_{u=1}^{n} \alpha\big(f_u(x)\big) \subseteq \bigcap_{v=1}^{m} S(g_v)$

(*ii*) $\alpha(y) \cup \bigcup_{v=1}^{m} \alpha\big(g_v(y)\big) \subseteq \bigcap_{u=1}^{n} S(f_u)$

Then we have for all $1 \leq u \leq n$ and $1 \leq v \leq m$:

$$f_u \circ g_v(x \parallel y) = f_u(x) \parallel g_v(y), \tag{1}$$

$$f_u \circ g_v(x \mathbin{⌊\!⌊} y) = f_u(x) \mathbin{⌊\!⌊} g_v(y). \tag{2}$$

Proof. First we will show that equation (1) is correct. With that result we will prove equation (2). Observe that this is not the "usual" order. This is caused by the fact that if we know that equation (2) holds, we do not know that

$$f_u \circ g_v(y \mathbin{⌊\!⌊} x) = g_v(y) \mathbin{⌊\!⌊} f_u(x).$$

We will prove (1) with induction to the sum n of the number of symbols of x and of y. First we will consider the basis of our induction: $n = 2$. We will have four possibilities:

$$x = a, y = b \quad x = a, y = \tau \quad x = \tau, y = b \quad x = y = \tau,$$

with $a, b \in A_\delta$. We will deduce only the first one. Let $1 \leq u \leq n$ and $1 \leq v \leq m$ be fixed. Consider the following.

$$\begin{aligned} f_u \circ g_v(a \parallel b) &= f_u \circ g_v(a \cdot b) + f_u \circ g_v(b \cdot a) \\ &= f_u(a) \cdot g_v(b) + g_v(b) \cdot f_u(a) \\ &= f_u(a) \parallel g_v(b). \end{aligned}$$

Now let $n \geq 2$, and suppose that (1) is correct for n. We will prove it for $n + 1$. Let x, y be chosen. Recall that they can be written as $BPA_{\delta,\tau}$-terms:

$$\begin{aligned} x &= \sum_i a_i \cdot x_i + \sum_k \tau \cdot t_k, \\ y &= \sum_j b_j \cdot y_j + \sum_l \tau \cdot s_l. \end{aligned}$$

Let u_i be such that $f_u(a_i \cdot z) = f_u(a_i) \cdot f_{u_i}(z)$ and let v_j be such that $g_v(b_j \cdot z) = g_v(b_j) \cdot g_{v_j}(z)$. Consider the calculation below.

$$\begin{aligned}
f_u \circ g_v(x \parallel y) &= \sum_i f_u \circ g_v\big(a_i \cdot (x_i \parallel y)\big) + \sum_k \tau \cdot f_u \circ g_v(t_k \parallel y) \\
&\quad + \sum_j f_u \circ g_v\big(b_j \cdot (x \parallel y_j)\big) + \sum_l \tau \cdot f_u \circ g_v(x \parallel s_l) \\
&= \sum_i f_u\big(a_i \cdot g_v(x_i \parallel y)\big) + \sum_k \tau \cdot f_u \circ g_v(t_k \parallel y) \\
&\quad + \sum_j f_u\big(g_v(b_j) \cdot g_{v_j}(x \parallel y_j)\big) + \sum_l \tau \cdot f_u \circ g_v(x \parallel s_l) \\
&= \sum_i f(a_i) \cdot f_{u_i} \circ g_v(x_i \parallel y) + \sum_k \tau \cdot f_u \circ g_v(t_k \parallel y) \\
&\quad + \sum_j g_v(b_j) \cdot f_u \circ g_{v_j}(x \parallel y_j) + \sum_l \tau \cdot f_u \circ g_v(x \parallel s_l)
\end{aligned}$$

It will be clear that we may use the induction hypothesis four times.

$$\begin{aligned}
&= \sum_i f(a_i) \cdot \big(f_{u_i}(x_i) \parallel g_v(y)\big) + \sum_k \tau \cdot \big(f_u(t_k) \parallel g_v(y)\big) \\
&\quad + \sum_j g_v(b_j) \cdot \big(g_{v_j}(y_j) \parallel f_u(x)\big) + \sum_l \tau \cdot \big(g_v(s_l) \parallel f_u(x)\big) \\
&= f_u(x) \mathbin{⫽} g_v(y) + g_v(y) \mathbin{⫽} f_u(x) \\
&= f_u(x) \parallel g_v(y).
\end{aligned}$$

This will end the proof of equation (1). To deduce equation (2) we find immediately that

$$f_u \circ g_v(x \mathbin{⫽} y) = \sum_i f_u(a_i) \cdot f_{u_i} \circ g_v(x_i \parallel y) + \sum_k \tau f_u \circ g_v(t_k \parallel y).$$

With the aid of equation (1), we see that this yields

$$\begin{aligned}
&= \sum_i f_u(a_i) \cdot \big(f_{u_i}(x_i) \parallel g_v(y)\big) + \sum_k \tau \cdot \big(f_u(t_k) \parallel g_v(y)\big) \\
&= f_u(x) \mathbin{⫽} g_v(y).
\end{aligned}$$

This will end the proof of (4.29).

Remark (4.30)

We treated theorem (4.29), as it can be seen as one of the first theorems concerning linear unary operators. It can be found in [**2**]. It is stated in terms of the so-called state operator and it uses the notion of the alphabet of an object to give the necessary conditions. This theorem, however, as it is stated in [**2**], is wrong. The definition of the alphabet of an object is "wrong". Even if we adjust this definition, the theorem still remains wrong. To exemplify this we will give hereinafter the definitions needed to formulate this theorem. Subsequently, we will state it and comment it. We will not have abstraction here, since it is not considered in [**2**]. The following definitions are taken from [**2**].

Definitions (4.31)

Let M and S be two given sets (with M finite), so that the sets A, M and S are pairwise disjoint. Suppose that two functions *act* and *eff* are given:

$$act : A \times M \times S \to A,$$
$$eff : A \times M \times S \to S.$$

We will write $a(m, s)$ for $act(a, m, s)$ and $s(m, a)$ for $eff(a, m, s)$. The defining axioms for the state operator will follow now.

(*i*)	$\lambda_s^m(\gamma) = \gamma$
(*ii*)	$\lambda_s^m(a) = a(m, s)$
(*iii*)	$\lambda_s^m(\gamma \cdot x) = \gamma \cdot \lambda_s^m(x)$
(*iv*)	$\lambda_s^m(a \cdot x) = a(m, s) \cdot \lambda_{s(m,a)}^m(x)$
(*v*)	$\lambda_s^m(x + y) = \lambda_s^m(x) + \lambda_s^m(y)$

Now we will give the definition of the alphabet $\alpha(m)$ of an object $m \in M$, as the set of all actions that can be changed, so

$$\alpha(m) = \{a \in A \mid \exists s \in S : a(m, s) \neq a\}.$$

The following is also copied from [**2**].

Theorem (4.32)

If there is no communication and $\alpha(x) \cap \alpha(m_1) = \alpha(y) \cap \alpha(m_2) = \emptyset$, then

$$\lambda_{s_1}^{m_1} \circ \lambda_{s_2}^{m_2}(x \parallel y) = \lambda_{s_1}^{m_1}(x) \parallel \lambda_{s_2}^{m_2}(y).$$

First of all we will show that this theorem is not correct. Let

$$M = \{m, m'\}, \quad S = \{s\}, \quad A = \{a, b, c, d\}.$$

We will give *act* and *eff*. We have $c(m, s) = d$ and $b(m', s) = c$. Further, nothing changes: $x(m, s) = x(m', s) = x$ for $x \in \{a, d\}$. Observe that $s(m, x) = s(m', x) = s$ for all $x \in A$. The conditions of (4.32) are satisfied for $x = a$ and $y = b$:

$$\alpha(a) \cap \alpha(m') = \{a\} \cap \{b\} = \emptyset,$$
$$\alpha(b) \cap \alpha(m) = \{b\} \cap \{c\} = \emptyset.$$

It is very easy to see that $\lambda_s^m \circ \lambda_s^{m'}(a \parallel b) = a \parallel d$, but we also see that $\lambda_s^m(a) \parallel \lambda_s^{m'}(b) = a \parallel c$.

We will explain what is "wrong" with $\alpha(m)$, for $m \in M$. If we have the situation that $a(m, s) = a$, we can still have $s(m, a) \neq s$. So this action has changed the operator. For the alphabet of an object $m \in M$ we take the following set:

$$\alpha(m) = \{a \in A \mid \exists s \in S : a(m, s) \neq a\} \cup \{a \in A \mid \exists t \in S : t(m, a) \neq t\}.$$

With this definition the example presented above, is still a counterexample for (4.32). Observe that we have the following:

$$\alpha(m) = \bigcup_{s \in S} A \setminus S(\lambda_s^m). \tag{3}$$

Now if we adjust the conditions in (4.32), as is done below, we have a correct formulation of this theorem.

Theorem (4.33)

Let x, y be closed *ACP*-terms. Suppose that there is no communication. If we have for m_1, m_2 in M and s_1, s_2 in S the following

$$\Big[\alpha(x) \cup \bigcup_{s \in S} \alpha\big(\lambda_s^{m_1}(x)\big)\Big] \cap \alpha(m_2) = \emptyset,$$
$$\Big[\alpha(y) \cup \bigcup_{s \in S} \alpha\big(\lambda_s^{m_2}(y)\big)\Big] \cap \alpha(m_1) = \emptyset,$$

then we have for $\star = \|, \mathbin{|\!\lfloor}$:

$$\lambda_{s_1}^{m_1} \circ \lambda_{s_2}^{m_2}(x \star y) = \lambda_{s_1}^{m_1}(x) \star \lambda_{s_2}^{m_2}(y).$$

Observe that the conditions here are more or less the same as in (4.29). In fact, the conditions in here are stronger, since in general

$$\big\{\, t \in S : \lambda_t^m \in D(\lambda_s^m) \,\big\} \subsetneq S.$$

With the aid of equation (3), we transform easily from the "empty intersection conditions" to the form in (4.29).

Proof. This is left to the reader.

5. A Model

In this section we will not construct a model for $ACP_{\tau,u}$, but we will "forget" about all the axioms concerning τs. Instead, we will confine ourselves to an axiom system that is called ACP_u. See table 5 on page 39. We will construct for ACP_u the standard model of process algebra: the projective limit model. We will do this by first making a row of finite models and after that constructing the projective limit. We will prove for each finite model that it makes ACP_u, *RDP*, *RSP*, *AIP*, *EA*, *ODP* and *OSP* true. Then we will prove the same items for the projective limit, using the results for the finite models. In fact, we show that the properties that are valid for the finite models are preserved under projective limits. This is a subject of research in model theory and is known under the name of "preservation theorems". In [**11**] we find in exercise 5.2.25* a preservation theorem concerning single-sorted projective limits. It says that a sentence φ is preserved under inverse limits (= projective limits) if and only if φ is equivalent to a sentence of the form

$$\bigwedge_{i=1}^{n} (\forall x_1, \ldots, x_s)\big((\forall y_1, \ldots, y_t)\psi_i \to \theta_i\big),$$

where ψ_i and θ_i are quantifier-free positive formulas. We will work out every proof since we have two-sorted algebras, but it is the the author's opinion, that it is worthwhile investigating many-sorted inverse (and direct) limits separately in connection to preservation theorems such as the one mentioned above. A general reference to many-sorted algebras is [**16**].

Definition (5.1)

Let G be the set of all closed terms over the theory T (see table 4). Let p be an element of G. We define the following subset of G.

$$[p]_n = \big\{\, q \in G : T \vdash \pi'_n(q) = \pi'_n(p) \,\big\}.$$

Now we define the set A_n to be the following.

$$A_n = \big\{\, [p]_n : p \in G \,\big\}.$$

* This is not a note: the asterisk belongs to the name of the exercise.

$x + y = y + x$ A1		$\partial'_H(a) = a, \quad \text{if } a \notin H$	D1′
$x + (y + z) = (x + y) + z$	A2	$\partial'_H(a) = \delta, \quad \text{if } a \in H$	D2′
$x + x = x$	A3	$\partial'_H(x \cdot y) = \partial'_H(x) \cdot \partial'_H(y)$	D3′
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4	$\partial'_H(x + y) = \partial'_H(x) + \partial'_H(y)$	D4′
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5		
$x + \delta = x$	A6	$\pi'_k(a) = a$	PR1′
$\delta \cdot x = \delta$	A7	$\pi'_1(a \cdot x) = a$	PR2′
		$\pi'_{k+1}(a \cdot x) = a \cdot \pi'_k(x)$	PR3′
$x \parallel y = x \mathbin{\|\!\lfloor} y + y \mathbin{\|\!\lfloor} x + x \mid y$	CM1	$\pi'_k(x + y) = \pi'_k(x) + \pi'_k(y)$	PR4′
$a \mathbin{\|\!\lfloor} x = a \cdot x$	CM2		
$(a \cdot x) \mathbin{\|\!\lfloor} y = a \cdot (x \parallel y)$	CM3	$a \mid b = b \mid a$	C1
$(x + y) \mathbin{\|\!\lfloor} z = x \mathbin{\|\!\lfloor} z + y \mathbin{\|\!\lfloor} z$	CM4	$(a \mid b) \mid c = a \mid (b \mid c)$	C2
$(a \cdot x) \mid b = (a \mid b) \cdot x$	CM5	$\delta \mid a = \delta$	C3
$a \mid (b \cdot x) = (a \mid b) \cdot x$	CM6		
$(a \cdot x) \mid (b \cdot y) = (a \mid b) \cdot (x \parallel y)$	CM7		
$(x + y) \mid z = x \mid z + y \mid z$	CM8		
$x \mid (y + z) = x \mid y + x \mid z$	CM9		

Table 4. An axiom system abbreviated by T.

Definition (5.2)

Let $\phi : A_n \longrightarrow A_n$ be a function. Suppose that we have for this function $\phi([\delta]_n) = [\delta]_n$, $\phi(x + y) = \phi(x) + \phi(y)$ for all $x, y \in A_n$ and for all $p \in G$ and l with $1 \le l \le n$ we have

$$\phi\big([p]_n\big) = [q]_n \Longrightarrow \phi\big(\big[\pi'_l(p)\big]_n\big) = \big[\pi'_l(q)\big]_n,$$

then we will call such a function a *laminal* function. We will use the abbreviation H_n for the set of all laminal functions.

Definition (5.3)

Let $p, q \in G$. We will define here some operators. First the binary operators, with both arguments in A_n.

$$[p]_n \star [q]_n = [p \star q]_n, \text{ for } \star = +, \cdot, \parallel, \mathbin{\|\!\lfloor}, \mid .$$

Secondly the unary operators with their argument in A_n. Let $k \ge 1$. Then we define the projection operator $\pi_k : A_n \longrightarrow A_n$ to be $\pi_k\big([p]_n\big) = \big[\pi'_k(p)\big]_n$. Now let H be a subset of the set of atomic actions, then we define the encapsulation operator $\partial_H : A_n \longrightarrow A_n$ to be $\partial_H\big([p]_n\big) = \big[\partial'_H(p)\big]_n$.

Lemma (5.4)

Let $x, y \in G$. Let $k, l \geq 1$. Let $H \subseteq A$. Then the following holds.

(*i*) $\pi'_k \circ \pi'_l(x) = \pi'_{\min(k,l)}(x)$

(*ii*) $\pi'_k \circ \partial'_H(x) = \partial'_H \circ \pi'_k(x)$

(*iii*) $\pi'_k(x \cdot y) = \pi'_k\big(\pi'_k(x) \cdot \pi'_k(y)\big)$

(*iv*) $\pi'_k(x \mid y) = \pi'_k\big(\pi'_k(x) \mid \pi'_k(y)\big)$

(*v*) $\pi'_k(x \mathbin{\|\!\_} y) = \pi'_k\big(\pi'_k(x) \mathbin{\|\!\_} \pi'_k(y)\big)$

(*vi*) $\pi'_k(x \parallel y) = \pi'_k\big(\pi'_k(x) \parallel \pi'_k(y)\big)$

Proof. Most of these properties have been proved in [**18**] for processes that can be defined with the aid of a guarded recursive specification. The proof of (*ii*) is completely analogous to the proof of theorem (4.8). This will end the proof of lemma (5.4).

Lemma (5.5)

The operations that we introduced in (5.3) are well-defined, i.e., the operators are independent of the choice of the representatives.

Proof. Suppose that $p' \in [p]_n$ and $q' \in [q]_n$. Then we obviously have:

$$\pi'_n(p') = \pi'_n(p), \text{ and } \pi'_n(q') = \pi'_n(q).$$

Consider the following calculation:

$$\begin{aligned} \pi'_n(p' + q') &= \pi'_n(p') + \pi'_n(q') \\ &= \pi'_n(p) + \pi'_n(q) \\ &= \pi'_n(p + q). \end{aligned}$$

Thus, we find that $[p' + q']_n = [p + q]_n$ and the alternative composition is independent of the choice of the representatives. Now let $\star = \cdot, \|, \mathbin{\|\!\_}$ or $|$. Then consider the following:

$$\begin{aligned} \pi'_n(p' \star q') &= \pi'_n\big(\pi'_n(p') \star \pi'_n(q')\big) \\ &= \pi'_n\big(\pi'_n(p) \star \pi'_n(q)\big) \\ &= \pi'_n(p \star q). \end{aligned}$$

Here, we make use of lemma (5.4). We will consider the unary operators that we introduced in (5.3). Let $p' \in [p]_n$. Then we have for $k \geq 1$:

$$\begin{aligned} \pi'_n \circ \pi'_k(p') &= \pi'_{\min(k,n)}(p') \\ &= \pi'_k \circ \pi'_n(p') \\ &= \pi'_k \circ \pi'_n(p) \\ &= \pi'_{\min(k,n)}(p) \\ &= \pi'_n \circ \pi'_k(p). \end{aligned}$$

Thus, we find $T \vdash \pi'_n \circ \pi'_k(p') = \pi'_n \circ \pi'_k(p)$. Let $H \subseteq A$ and $p' \in [p]_n$. Then, we have the following.

$$\begin{aligned} \pi'_n \circ \partial'_H(p') &= \partial'_H \circ \pi'_n(p') \\ &= \partial'_H \circ \pi'_n(p) \\ &= \pi'_n \circ \partial'_H(p). \end{aligned}$$

So we see that $T \vdash \pi'_n \circ \partial'_H(p') = \pi'_n \circ \partial'_H(p)$; thus we find that $\partial_H\big([p]_n\big) = \partial_H\big([p']_n\big)$. And we see that all the operators that we introduced in (5.3) are well-defined. This ends the proof of our lemma.

Lemma (5.6)

Let $n \geq 1$. For all $k \geq 1$ we have $\pi_k \in H_n$. For all $H \subseteq A$ we have $\partial_H \in H_n$.

Proof. We are to show for these operators that they are laminal functions. For the definition of a laminal function, see (5.2). Let $n, k \geq 1$. It is evidently clear, that π_k is the identity on $[\delta]_n$. The same is valid for ∂_H, for every subset $H \subseteq A$. Let $[p]_n, [q]_n \in A_n$ and let ϕ be the projection operator, or the encapsulation operator; and let ϕ' be the acuted corresponding operator. Then consider the following.

$$\begin{aligned}\phi\big([p]_n + [q]_n\big) &= \phi\big([p+q]_n\big)\\ &= \big[\phi'(p+q)\big]_n\\ &= \big[\phi'(p)\big]_n + \big[\phi'(q)\big]_n\\ &= \phi\big([p]_n\big) + \phi\big([q]_n\big).\end{aligned}$$

It follows immediately from lemma (5.4)(i) that π_k is a laminal function. It follows at once from lemma (5.4)(ii) that ∂_H is a laminal function. This will end the proof of lemma (5.6).

Remark (5.7)

We know that $|H_n| < \infty$, so $\{\pi_k : k \geq 1\}$ must be finite. Let $[p]_n \in A_n$. Because of lemma (5.4), we can find the following for all $k \geq 0$:

$$\pi'_{n+k} \circ \pi'_n(p) = \pi'_n(p).$$

Thus, we find $\pi_{n+k}\big([p]_n\big) = \pi_n\big([p]_n\big)$ for all $k \geq 0$ and $[p]_n \in A_n$. Moreover, we see that π_{n+k} is the identity map for all $k \geq 0$.

Definition (5.8)

We define $\chi : H_n \times A_n \longrightarrow A_n$ as follows: $\chi(f, x) = f(x)$, for $f \in H_n$ and $x \in A_n$. We define the composition of functions $\circ : H_n \times H_n \longrightarrow H_n$ as follows. Let $(f, g) \in H_n \times H_n$, then $f \circ g : A_n \longrightarrow A_n$ is defined: $f \circ g(x) = f\big(g(x)\big)$, for $x \in A_n$. It is easy to see that $f \circ g$ is indeed a laminal function.

Now we are in a position to give the following definition.

Definition (5.9)

Let $\mathfrak{M}_n$ be the algebra that consists of the sets A_n and H_n, the operators $+, \cdot, \|, \mathbin{\|\!\_}, |, \chi, \circ$ and the constants $[a]_n \in A_n$ for all $a \in A$ and the constants $\pi_k, \partial_H \in H_n$ for every $k \geq 1$ and $H \subseteq A$.

Theorem (5.10)

Let $n \geq 1$, then we have $\mathfrak{M}_n \models ACP_u$. See table 5 on page 39 for the axiom system ACP_u.

Proof. We are to show that $\mathfrak{M}_n$ models each axiom of ACP_u. We will only sketch the proof, since the other axioms are deduced in the same way as the examples below are proved. We will show that $\mathfrak{M}_n \models$ A4.

$$\begin{aligned}\big([p]_n + [q]_n\big) \cdot [r]_n &= \big([p+q]_n\big) \cdot [r]_n\\ &= \big[(p+q) \cdot r\big]_n\\ &= [p \cdot r + q \cdot r]_n\\ &= [p \cdot r]_n + [q \cdot r]_n\\ &= [p]_n \cdot [r]_n + [q]_n \cdot [r]_n.\end{aligned}$$

$x + y = y + x$	A1	$\partial_H(a) = a, \quad \text{if } a \notin H$	D1
$x + (y + z) = (x + y) + z$	A2	$\partial_H(a) = \delta, \quad \text{if } a \in H$	D2
$x + x = x$	A3	$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	D3
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4		
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5	$\pi_n(a) = a$	PR1
$x + \delta = x$	A6	$\pi_1(a \cdot x) = a$	PR2
$\delta \cdot x = \delta$	A7	$\pi_{n+1}(a \cdot x) = a \cdot \pi_n(x)$	PR3
$x \parallel y = x \mathbin{⌊\!⌊} y + y \mathbin{⌊\!⌊} x + x \mid y$	CM1	$a \mid b = b \mid a$	C1
$a \mathbin{⌊\!⌊} x = a \cdot x$	CM2	$(a \mid b) \mid c = a \mid (b \mid c)$	C2
$(a \cdot x) \mathbin{⌊\!⌊} y = a \cdot (x \parallel y)$	CM3	$\delta \mid a = \delta$	C3
$(x + y) \mathbin{⌊\!⌊} z = x \mathbin{⌊\!⌊} z + y \mathbin{⌊\!⌊} z$	CM4		
$(a \cdot x) \mid b = (a \mid b) \cdot x$	CM5	$\chi(f \circ g, x) = \chi\big(f, \chi(g, x)\big)$	XC1
$a \mid (b \cdot x) = (a \mid b) \cdot x$	CM6	$\chi\big((f \circ g) \circ h, x\big) = \chi\big(f \circ (g \circ h), x\big)$	XC2
$(a \cdot x) \mid (b \cdot y) = (a \mid b) \cdot (x \parallel y)$	CM7	$\chi(f, \gamma) = \gamma$	X1
$(x + y) \mid z = x \mid z + y \mid z$	CM8	$\chi(f, \gamma \cdot x) = \gamma \cdot \chi(f, x)$	X2
$x \mid (y + z) = x \mid y + x \mid z$	CM9	$\chi(f, x + y) = \chi(f, x) + \chi(f, y)$	X3

Table 5. ACP_u.

Now we will infer $\mathfrak{M}_n \models$ CM7.

$$\begin{aligned}
\big([a]_n \cdot [p]_n\big) \bigm| \big([b]_n \cdot [q]_n\big) &= [a \cdot p]_n \bigm| [b \cdot q]_n \\
&= \big[(a \cdot p) \mid (b \cdot q)\big]_n \\
&= \big[(a \mid b) \cdot (p \parallel q)\big]_n \\
&= [a \mid b]_n \cdot [p \parallel q]_n \\
&= \big([a]_n \bigm| [b]_n\big) \cdot \big([p]_n \bigm\| [q]_n\big).
\end{aligned}$$

We will prove $\mathfrak{M}_n \models$ D3.

$$\begin{aligned}
\chi\big(\partial_H, [p]_n + [q]_n\big) &= \chi\big(\partial_H, [p + q]_n\big) \\
&= \partial_H\big([p + q]_n\big) \\
&= \big[\partial'_H(p + q)\big]_n \\
&= \big[\partial'_H(p) + \partial'_H(q)\big]_n \\
&= \big[\partial'_H(p)\big]_n + \big[\partial'_H(q)\big]_n \\
&= \partial_H\big([p]_n\big) + \partial_H\big([q]_n\big) \\
&= \chi\big(\partial_H, [p]_n\big) + \chi\big(\partial_H, [q]_n\big).
\end{aligned}$$

We will show $\mathfrak{M}_n \models$ PR3. Let $k \geq 1$.

$$\begin{aligned}
\chi\big(\pi_{k+1}, [a]_n \cdot [p]_n\big) &= \pi_{k+1}\big([a \cdot p]_n\big) \\
&= \big[\pi'_{k+1}(a \cdot p)\big]_n \\
&= \big[a \cdot \pi'_k(p)\big]_n \\
&= [a]_n \cdot \big[\pi'_k(p)\big]_n \\
&= [a]_n \cdot \pi_k\big([p]_n\big) \\
&= [a]_n \cdot \chi\big(\pi_k, [p]_n\big).
\end{aligned}$$

Now we will prove XC1 and X2. Suppose that $h, k \in H_n$ and $x \in A_n$.

$$\begin{aligned}
\chi(h \circ k, x) &= h \circ k(x) \\
&= \chi\big(h, k(x)\big) \\
&= \chi\big(h, \chi(k, x)\big).
\end{aligned}$$

Let $h \in H_n$ and let $[p]_n \in A_n$.

$$\begin{aligned}
\chi\big(h, [\delta]_n \cdot [p]_n\big) &= h\big([\delta]_n \cdot [p]_n\big) \\
&= h\big([\delta]_n\big) \\
&= [\delta]_n \\
&= \big[\delta \cdot \chi\big(h, [p]_n\big)\big]_n \\
&= [\delta]_n \cdot \chi\big(h, [p]_n\big).
\end{aligned}$$

This ends the sketch of the proof of (5.10).

$(x+y)+z \to x+(y+z)$	RA2	$\partial'_H(a) \to a, \quad$ if $a \notin H$	RD1′
$x+x \to x$	RA3	$\partial'_H(a) \to \delta, \quad$ if $a \in H$	RD2′
$(x+y)\cdot z \to x\cdot z + y \cdot z$	RA4	$\partial'_H(x \cdot y) \to \partial'_H(x) \cdot \partial'_H(y)$	RD3′
$(x \cdot y)\cdot z \to x \cdot (y \cdot z)$	RA5	$\partial'_H(x+y) \to \partial'_H(x) + \partial'_H(y)$	RD4′
$x + \delta \to x$	RA6		
$\delta \cdot x \to \delta$	RA7	$\pi'_k(a) \to a$	RPR1′
		$\pi'_1(a \cdot x) \to a$	RPR2′
$x \parallel y \to x \mathbin{⌊\!⌊} y + (y \mathbin{⌊\!⌊} x + x \mid y)$	RCM1	$\pi'_{k+1}(a \cdot x) \to a \cdot \pi'_k(x)$	RPR3′
$a \mathbin{⌊\!⌊} x \to a \cdot x$	RCM2	$\pi'_k(x+y) \to \pi'_k(x) + \pi'_k(y)$	RPR4′
$(a \cdot x) \mathbin{⌊\!⌊} y \to a \cdot (x \parallel y)$	RCM3		
$(x+y) \mathbin{⌊\!⌊} z \to x \mathbin{⌊\!⌊} z + y \mathbin{⌊\!⌊} z$	RCM4	$a \mid b \to c_{a,b}$	RC
$(a \cdot x) \mid b \to (a \mid b) \cdot x$	RCM5		
$a \mid (b \cdot x) \to (a \mid b) \cdot x$	RCM6		
$(a \cdot x) \mid (b \cdot y) \to (a \mid b) \cdot (x \parallel y)$	RCM7		
$(x+y) \mid z \to x \mid z + y \mid z$	RCM8		
$x \mid (y+z) \to x \mid y + x \mid z$	RCM9		

Table 6. A term rewriting system associated with the theory T.

Lemma (5.11)

The term rewriting system of table 6 has the termination property*.

Proof. We will prove (5.11) in the same way as we proved (3.14). Firstly, we will give the partial ordering of the signature. We will also use the ranked operators that we introduced in section 3.

$$\|_n > \mathbin{\|\!\lfloor}_n, |_n > \|_{n-1}, \quad \|_n, \mathbin{\|\!\lfloor}_n, |_n > \cdot, \quad \partial_H, \cdot > +, \quad \pi_{n+1} > \pi_n > \cdot, \quad |_2 > A > \delta.$$

We will use the lexicographical variant of the recursive path ordering, although we will need the lexicographical status only for the alternative composition and for the sequential composition. We are to show that for each rewriting rule $s \to t$ in table 6, $s \succ t$. For the symbol "$\succ$" we refer to definition (3.9). The treatment of the cases RA2–RA7, RCM1–RCM9 and RC, is the same as in theorem (3.14). The calculation of RPR1′ is trivial. Let us treat RPR2′.

$$\begin{aligned} \pi'_1(a \cdot x) &\succ \pi'^*_1(a \cdot x) \\ &\succ a \cdot x \\ &\succ a \cdot^* x \\ &\succ a. \end{aligned}$$

Now we will handle RPR3′. Let $k \geq 1$.

$$\begin{aligned} \pi'_{k+1}(a \cdot x) &\succ \pi'^*_{k+1}(a \cdot x) \\ &\succ \pi'^*_{k+1}(a \cdot x) \cdot \pi'^*_{k+1}(a \cdot x) \\ &\succ (a \cdot x) \cdot \pi'_k\big(\pi'^*_{k+1}(a \cdot x)\big) \\ &\succ (a \cdot^* x) \cdot \pi'_k(a \cdot x) \\ &\succ a \cdot \pi'^*_k(a \cdot x) \\ &\succ a \cdot \pi'_k(a \cdot^* x) \\ &\succ a \cdot \pi'_k(x). \end{aligned}$$

Now we will discuss RPR4′. Let $k \geq 1$.

$$\begin{aligned} \pi'_k(x + y) &\succ \pi'^*_k(x + y) \\ &\succ \pi'_k(x + y) +^* \pi'_k(x + y) \\ &\succ \pi'^*_k(x + y) + \pi'^*_k(x + y) \\ &\succ \pi'_k(x +^* y) + \pi'_k(x +^* y) \\ &\succ \pi'_k(x) + \pi'_k(y). \end{aligned}$$

RD1–RD4 are proved in the same way as in theorem (3.14), albeit that we use ∂'_H in the partial ordering instead of χ. This ends the proof of (5.11).

Lemma (5.12)

Let $n \geq 1$ be chosen. Let $p \in G$ be a closed term over the theory T. See (5.1) for the definition of G. Then there is an element $q \in G$, such that $[q]_n = [p]_n$ and $T \vdash \pi'_n(q) = q$. We will call q the n-normal form of p.

Proof. Since $p \in G$ is a closed T-term, $\pi'_n(p)$ is also a closed T-term. With the aid of lemma (5.11), we can rewrite this term to a term $q \in G$, which is in normal form. We know that $T \vdash q = \pi'_n(p)$, thus,

$$\begin{aligned} \pi'_n(q) &= \pi'_n\big(\pi'_n(p)\big) \\ &= \pi'_n(p) \\ &= q, \end{aligned}$$

with the aid of lemma (5.4) and we also see that $[q]_n = [p]_n$. This ends the proof of (5.12).

* See definition (3.11)

Lemma (5.13)

Let $p', q' \in G$ be closed terms and let p and q be their n-normal forms. Then we have the following.

$$\mathfrak{M}_n \models p = q \iff T \vdash p = q.$$

Proof. Let us assume that $\mathfrak{M}_n \models p = q$, with p and q n-normal forms. Then we find that $T \vdash \pi'_n(p) = \pi'_n(q)$. But because of lemma (5.12), we have $\pi'_n(p) = p$ and $\pi'_n(q) = q$. So we find $T \vdash p = q$.

Now let us assume that $T \vdash p = q$. Then we have a fortiori $T \vdash \pi'_n(p) = \pi'_n(q)$. So $[p]_n = [q]_n$ and we find $\mathfrak{M}_n \models p = q$. This ends the proof of (5.13).

Epiphenomenon (5.14)

Let $a_i = |A_i|$ be the number of elements of the set A_i that we defined in (5.1) ($i \geq 1$) and let $a_0 = 0$. Let $u = |A|$. Then we have the following recursive formula.

$$\begin{aligned} a_0 &= 0, \\ a_{i+1} &= 2^{u\cdot(1+a_i)}. \end{aligned}$$

Proof. We will give a sketch of the proof. Let $i = 1$. Because of axioms A3 and A6, we can make the following different sums:

$$1 + \sum_{i=1}^{u} \binom{u}{i} = 1 + (2^u - 1) = 2^u = 2^{u\cdot(1+a_0)}.$$

Let $i > 1$. We will have $u + u \cdot a_i$ terms of which the first symbol (in prefix notation) is not a sum. Thus, we obtain, just as above for $i = 1$, $2^{u\cdot(1+a_i)}$ possibilities for the number of different sums that we can make with these $u \cdot (1 + a_i)$ terms of which the sums are built up.

Examples (5.15)

In table 7 we will give some figures that we have calculated with the aid of the formula of (5.14). For instance, if we have two atomic actions, there are approximately $1.2 \cdot 10^{617}$ elements in A_3.

u	1	2	3	4	5	6
a_1	2	4	8	16	32	64
a_2	8	1024	$1.3 \cdot 10^8$	$2.9 \cdot 10^{20}$	$4.6 \cdot 10^{49}$	$2.5 \cdot 10^{117}$
a_3	512	$1.2 \cdot 10^{617}$	$-*$	$-$	$-$	$-$
a_4	$2.6 \cdot 10^{154}$	$-$	$-$	$-$	$-$	$-$

* ($> 10^{9999}$)

Table 7. Some calculations with (5.14).

Remark (5.16)

Let x be a variable. Let $C[x]$ be a context of x, in which the occurrence of x is guarded. Then we have for all $k \geq n$:

$$\pi'_{n+1}\big(C[x]\big) = \pi'_{n+1}\big(C\big[\pi'_k(x)\big]\big).$$

Theorem (5.17)

Let $n \geq 1$, then we have $\mathfrak{M}_n \models RDP, RSP$.

Proof. Let $V = \{x_\alpha : \alpha \in I\}$ be a set of variables. Let

$$E(V) = \left\{ x_\alpha = t_\alpha\big((x_\beta)_{\beta\in I}\big) : \alpha \in I \right\}$$

be a guarded recursive specification. Without loss of generality, we may assume that $E(V)$ is completely guarded; see definition (2.15). We will prove (5.17) with induction to n. Thus, let $n = 1$. For all $\alpha \in I$, we calculate $p^1_\alpha := \pi'_1(x_\alpha)$. This is a closed term over the theory T.

$$\begin{aligned} \pi'_1\big(t_\alpha\big((p^1_\beta)_\beta\big)\big) &= \pi'_1\big(t_\alpha\big((x_\beta)_\beta\big)\big) \\ &= \pi'_1(x_\alpha) \\ &= \pi'_1 \circ \pi'_1(x_\alpha) \\ &= \pi'_1(p^1_\alpha). \end{aligned} \tag{1}$$

In (1) we use the fact that $\pi'_1(a \cdot x) = \pi'_1(a \cdot y)$ for all x and y in combination with the fact that $E(V)$ is completely guarded. Henceforward, we will use this argument tacitly. So we see that $(p^1_\alpha)_\alpha$ solves the system $E(V)$ in $\mathfrak{M}_1$. Now suppose that $(q_\alpha)_\alpha$ also solves this system, i.e., $\pi'_1(q_\alpha) = \pi'_1\big(t_\alpha\big((q_\beta)_\beta\big)\big)$. Then we find the following:

$$\begin{aligned} \pi'_1(q_\alpha) &= \pi'_1\big(t_\alpha\big((q_\beta)_\beta\big)\big) \\ &= \pi'_1\big(t_\alpha\big((x_\beta)_\beta\big)\big) \\ &= \pi'_1(x_\alpha) \\ &= \pi'_1 \circ \pi'_1(x_\alpha) \\ &= \pi'_1(p^1_\alpha). \end{aligned}$$

We see that the solution $(q_\alpha)_\alpha$ is just the one that we have already constructed, so $(p^1_\alpha)_\alpha$ is the unique solution. Now we see that (5.17) is correct for $n = 1$. Let $n \geq 1$ and suppose that (5.17) has been proved for n. Let $(p^n_\alpha)_\alpha$ be such that

$$T \vdash \pi'_n(p^n_\alpha) = \pi'_n\big(t_\alpha\big((p^n_\beta)_\beta\big)\big).$$

Define $p^{n+1}_\alpha := t_\alpha\big((p^n_\beta)_\beta\big)$. We will show that

$$\pi'_{n+1}(p^{n+1}_\alpha) = \pi'_{n+1}\big(t_\alpha\big((p^{n+1}_\beta)_\beta\big)\big).$$

Consider thereunto the following.

$$\begin{aligned} \pi'_{n+1}(p^{n+1}_\alpha) &= \pi'_{n+1}\big(t_\alpha\big((p^n_\beta)_\beta\big)\big) \\ &= \pi'_{n+1}\Big(t_\alpha\big(\big(\pi'_n(p^n_\beta)\big)_\beta\big)\Big) \\ &= \pi'_{n+1}\Big(t_\alpha\big(\pi'_n\big(t_\beta\big((p^n_\gamma)_\gamma\big)\big)_\beta\big)\Big) \\ &= \pi'_{n+1}\Big(t_\alpha\big(t_\beta\big((p^n_\gamma)_\gamma\big)_\beta\big)\Big) \\ &= \pi'_{n+1}\big(t_\alpha\big((p^{n+1}_\beta)_\beta\big)\big). \end{aligned} \tag{2}$$

In equation (2), we use the fact that $E(V)$ is completely guarded and we use remark (5.16). In the sequel of this proof, we will apply this argument tacitly. We find that $(p^{n+1}_\alpha)_\alpha$ solves $E(V)$ in $\mathfrak{M}_{n+1}$. Now we will show that this solution is unique. Let $(q_\alpha)_\alpha$ be such that

$$T \vdash \pi'_{n+1}(q_\alpha) = \pi'_{n+1}\big(t_\alpha\big((q_\beta)_\beta\big)\big).$$

Observe that the following holds.

$$\begin{aligned}\pi'_n(p_\alpha^{n+1}) &= \pi'_n\big(t_\alpha\big((p_\beta^n)_\beta\big)\big)\\ &= \pi'_n(p_\alpha^n).\end{aligned} \tag{3}$$

First, we will show that $\pi'_n(q_\alpha)$ is a solution in $\mathfrak{M}_n$.

$$\begin{aligned}\pi'_n(q_\alpha) &= \pi'_n \circ \pi'_{n+1}(q_\alpha)\\ &= \pi'_n \circ \pi'_{n+1}\big(t_\alpha\big((q_\beta)_\beta\big)\big)\\ &= \pi'_n\big(t_\alpha\big((q_\beta)_\beta\big)\big).\end{aligned}$$

But in $\mathfrak{M}_n$ the solution is unique, so we find with (3) that

$$\pi'_n(q_\alpha) = \pi'_n(p_\alpha) = \pi'_n(p_\alpha^{n+1}).$$

Now we will show that $\pi'_{n+1}(q_\alpha) = \pi'_{n+1}(p_\alpha^{n+1})$.

$$\begin{aligned}\pi'_{n+1}(q_\alpha) &= \pi'_{n+1}\big(t_\alpha\big((q_\beta)_\beta\big)\big)\\ &= \pi'_{n+1}\Big(t_\alpha\big(\big(\pi'_n(q_\beta)\big)_\beta\big)\Big)\\ &= \pi'_{n+1}\Big(t_\alpha\big(\big(\pi'_n(p_\beta^{n+1})\big)_\beta\big)\Big)\\ &= \pi'_{n+1}\big(t_\alpha\big((p_\beta^{n+1})_\beta\big)\big)\\ &= \pi'_{n+1}(p_\alpha^{n+1}).\end{aligned}$$

This ends the proof of the theorem.

Theorem (5.18)

Let $n \geq 1$, then $\mathfrak{M}_n \models AIP$.

Proof. Let $x, y \in A_n$. Suppose that for all $k \geq 1$, we have $\pi_k(x) = \pi_k(y)$. In particular we have this for $k = n$. With the aid of remark (5.7), we find that $\pi_n = id$, so we see that $x = y$. This will end the proof of (5.18).

Lemma (5.19)

Let $[p]_n \in A_n$. Then there are $a_1, \ldots, a_s, b_1, \ldots, b_t \in A \cup \{\delta\}$ and $p_1, \ldots, p_s \in G$, such that

$$[p]_n = \sum_{j=1}^{s} [a_j]_n \cdot [p_j]_n + \sum_{k=1}^{t} [b_k]_n.$$

Proof. Let p be a closed term over the theory T. See table 4 on page 36. If we take a close look at this axiom system, we see that this is in fact ACP with projections. From this system we know that there are unique normal forms for all closed terms (see [**5**]). They have the desired form:

$$p = \sum_{k=1}^{s} a_j \cdot p_j + \sum_{k=1}^{t} b_k,$$

for certain $a_1, \ldots, a_s, b_1, \ldots, b_t \in A \cup \{\delta\}$ and closed terms $p_1, \ldots, p_s$. Thus, we find

$$[p]_n = \sum_{j=1}^{s} [a_j]_n \cdot [p_j]_n + \sum_{k=1}^{t} [b_k]_n.$$

This ends the proof of the lemma.

Theorem (5.20)

Let $n \geq 1$, then $\mathfrak{M}_n \models EA$. See (2.1) for the definition of EA.

Proof. Since H_n is defined as a function space, we automatically have that for all $f, g \in H_n : f = g$, if and only if we have for all $x \in A_n : f(x) = g(x)$. This is what we wanted to prove.

Theorem (5.21)

Let $n \geq 1$, then $\mathfrak{M}_n \models ODP, OSP$.

Proof. Fix an $n \geq 1$. We will use the more explicit formulation of the definition of a linear functional specification that we have already described in (2.3). Let $M = \{m_1, \ldots, m_l\}$ be a set of function names and let $\sigma : A \times \{1, \ldots, l\} \longrightarrow \{1, \ldots, l\}$ be a given map. Recall that the linear functional specification $E(M)$ has the following form:

$$\begin{aligned} E(M) = \big\{ \, m_i(a) = a^{(i)} : a \in A, \; 1 \leq i \leq l \, \big\} & \\ \cup \big\{ \, m_i(a \cdot x) = a^{(i)} \cdot m_{\sigma(a,i)}(x) : a \in A, \; 1 \leq i \leq l \, \big\}&. \end{aligned}$$

First we will prove that there is a valuation $\varphi : M \longrightarrow H_n$, which solves the system of equations $E(M)$. See section 2 for the definition of a valuation. It suffices to show that there are $\mu_1, \ldots, \mu_l \in H_n$ such that

$$\mu_i\big([a]_n\big) = \big[a^{(i)}\big]_n, \tag{4}$$

$$\mu_i\big([a]_n \cdot [p]_n\big) = \big[a^{(i)}\big]_n \cdot \mu_{\sigma(a,i)}\big([p]_n\big). \tag{5}$$

(So we can take $\varphi(m_i) := \mu_i$.) Let $[p]_n \in A_n$, then there are $a_1, \ldots, a_s, b_1, \ldots, b_t \in A \cup \{\delta\}$ and $p_1, \ldots, p_s \in G$ such that

$$[p]_n = \sum_{j=1}^{s} [a_j]_n \cdot [p_j]_n + \sum_{k=1}^{t} [b_k]_n, \tag{6}$$

according to lemma (5.19). Let $\mu_i^1 : A_n \longrightarrow A_n$ be defined as follows:

$$\begin{aligned} \mu_i^1\big([\delta]_n\big) &= [\delta]_n, \\ \mu_i^1\big([p]_n\big) &= \sum_{j=1}^{s} \big[a_j^{(i)}\big]_n + \sum_{k=1}^{t} \big[b_k^{(i)}\big]_n. \end{aligned}$$

It consists of straightforward calculation to show that $\mu_i^1 \in H_n$. Let $1 \leq r < n$. Suppose that $\mu_i^r \in H_n$, for all $1 \leq i \leq l$, then we define

$$\mu_i^{r+1} : A_n \longrightarrow A_n$$

to be the identity on $[\delta]_n$ and for $[p]_n \in A_n$

$$\mu_i^{r+1}\big([p]_n\big) = \sum_{j=1}^{s} \big[a_j^{(i)}\big]_n \cdot \mu^r_{\sigma(a_j,i)}\big([p_j]_n\big) + \sum_{k=1}^{t} \big[b_k^{(i)}\big]_n.$$

We will show that $\mu_i^{r+1} \in H_n$. It is trivial to prove that μ_i^{r+1} distributes over the alternative composition. Let $[p]_n \in A_n$ be as in equation (6). Let $\mu^r_{\sigma(a_j,i)}\big([p_j]_n\big) = [q_j]_n$. Furthermore, let

$$q = \sum_{j=1}^{s} a_j^{(i)} \cdot q_j + \sum_{k=1}^{t} b_k^{(i)}.$$

Then we see that $\mu_i^{r+1}\big([p]_n\big) = [q]_n$. We will prove for all v, with $1 \le v \le n$ that

$$\mu_i^{r+1}\big(\big[\pi'_v(p)\big]_n\big) = \big[\pi'_v(q)\big]_n. \tag{7}$$

If $v = 1$, we see immediately that equation (7) is valid. Now suppose that $1 < v \le n$, then consider the subsequent deduction.

$$\begin{aligned}
\mu_i^{r+1}\big(\big[\pi'_v(p)\big]_n\big) &= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \mu^r_{\sigma(a_j,i)}\big(\big[\pi'_{v-1}(p_j)\big]_n\big) + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n \\
&= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \big[\pi'_{v-1}(q_j)\big]_n + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n \\
&= \big[\pi'_v(q)\big]_n.
\end{aligned}$$

This shows that μ_i^{r+1} is a laminal function. We will prove a technical result on these laminal functions.

$$\pi_{n-r} \circ \mu_i^{n-r} = \pi_{n-r} \circ \mu_i^{n-r+1}, \quad 1 \le i \le l,\ 1 \le r \le n-1. \tag{8}$$

We will prove (8) first for $r = n-1$. Let $[p]_n \in A_n$ be as in equation (6). Consider the calculation hereinafter.

$$\begin{aligned}
\pi_1 \circ \mu_i^2\big([p]_n\big) &= \pi_1\Big(\sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \mu^1_{\sigma(a_j,i)}\big([p_j]_n\big) + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n\Big) \\
&= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_n + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n \\
&= \pi_1\Big(\sum_{j=1}^{s}\big[a_j^{(i)}\big]_n + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n\Big) \\
&= \pi_1 \circ \mu_i^1\big([p]_n\big).
\end{aligned}$$

So for $r = n-1$ we see that (8) is correct. Suppose that (8) has been verified for $1 < r \le n-1$. We will prove it for $r-1$. Let $[p]_n \in A_n$ be as in equation (6). Consider the following.

$$\begin{aligned}
\pi_{n-r+1} \circ \mu_i^{n-r+1}\big([p]_n\big) &= \pi_{n-r+1}\Big(\sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \mu^{n-r}_{\sigma(a_j,i)}\big([p_j]_n\big) + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n\Big) \\
&= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \pi_{n-r} \circ \mu^{n-r}_{\sigma(a_j,i)}\big([p_j]_n\big) + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n \\
&= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \pi_{n-r} \circ \mu^{n-r+1}_{\sigma(a_j,i)}\big([p_j]_n\big) + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n \\
&= \pi_{n-r+1}\Big(\sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \mu^{n-r+1}_{\sigma(a_j,i)}\big([p_j]_n\big) + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n\Big) \\
&= \pi_{n-r+1} \circ \mu_i^{n-r+2}\big([p]_n\big).
\end{aligned}$$

This will end the verification of (8). In particular, we find that (8) is valid for $r = 1$. Thus, we obtain the following formula:

$$\pi_{n-1} \circ \mu_i^{n-1} = \pi_{n-1} \circ \mu_i^{n}. \tag{9}$$

Define $\mu_i = \mu_i^n \in H_n$ for $1 \leq i \leq l$. We claim that these functions satisfy equations (4) and (5). It will be clear that

$$\mu_i\big([a]_n\big) = \big[a^{(i)}\big]_n,$$

for all $1 \leq i \leq l$, so we see that $\mu_1, \dots, \mu_l$ satisfy the boundary conditions of $E(M)$, i.e., equation (4). Now let us take a look at the functional equations of $E(M)$, or equivalently, equation (5). Let $i \in \{1, \dots, l\}$ be fixed.

$$\begin{aligned}
\mu_i\big([a]_n \cdot [p]_n\big) &= \mu_i^n\big([a]_n \cdot [p]_n\big) \\
&= \pi_n \circ \mu_i^n\big([a]_n \cdot [p]_n\big) && \text{see remark (5.7)} \\
&= \pi_n\Big(\big[a^{(i)}\big]_n \cdot \mu_{\sigma(a,i)}^{n-1}\big([p]_n\big)\Big) \\
&= \big[a^{(i)}\big]_n \cdot \pi_{n-1} \circ \mu_{\sigma(a,i)}^{n-1}\big([p]_n\big) \\
&= \big[a^{(i)}\big]_n \cdot \pi_{n-1} \circ \mu_{\sigma(a,i)}^{n}\big([p]_n\big) && \text{because of (9)} \\
&= \pi_n\Big(\big[a^{(i)}\big]_n \cdot \mu_{\sigma(a,i)}^{n}\big([p]_n\big)\Big) \\
&= \big[a^{(i)}\big]_n \cdot \mu_{\sigma(a,i)}\big([p]_n\big).
\end{aligned}$$

This will end the proof of the existential part of (5.21). Now we will prove the uniqueness part: $\mathfrak{M}_n \models \mathit{OSP}$. Suppose that there is also a valuation $\phi : M \longrightarrow H_n$, which solves the system of equations $E(M)$. Let $\phi(m_i) := \nu_i$ for all i. Then we have for all $i \in \{1, \dots, l\}$

$$\begin{aligned}
\nu_i\big([a]_n\big) &= \big[a^{(i)}\big]_n, \\
\nu_i\big([a]_n \cdot [p]_n\big) &= \big[a^{(i)}\big]_n \cdot \nu_{\sigma(a,i)}\big([p]_n\big).
\end{aligned}$$

In order to verify that $\mu_i = \nu_i$ (for all i), we will prove the following:

$$\pi_r \circ \nu_i = \mu_i^r, \quad 1 \leq r \leq n,\ 1 \leq i \leq l. \tag{10}$$

We will prove (10) with induction to r. Let $[p]_n \in A_n$ be as in equation (6). Let $r = 1$ and let $i \in \{1, \dots, l\}$, then we see the following

$$\begin{aligned}
\pi_1 \circ \nu_i\big([p]_n\big) &= \pi_1\Big(\sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \nu_{\sigma(a_j,i)}\big([p_j]_n\big) + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n\Big) \\
&= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_n + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n \\
&= \mu_i^1\big([p]_n\big).
\end{aligned}$$

Thus, (10) is correct for $r = 1$. Now suppose that (10) is valid for $1 \leq r < n$, then we prove it for $r + 1$. Let $[p]_n \in A_n$ be as in equation (6).

$$\begin{aligned}
\pi_{r+1} \circ \nu_i\big([p]_n\big) &= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \pi_r \circ \nu_{\sigma(a_j,i)}\big([p_j]_n\big) + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n \\
&= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_n \cdot \mu_{\sigma(a_j,i)}^{r}\big([p_j]_n\big) + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n \\
&= \mu_i^{r+1}\big([p]_n\big).
\end{aligned}$$

This proves (10). In particular we find that (10) is valid for $r = n$. If we combine this with the fact that π_n is the identity map—see remark (5.7)—we find for all $1 \leq i \leq l$ that $\nu_i = \pi_n \circ \nu_i = \mu_i^n = \mu_i$. This will end the verification of the uniqueness part and therewith the proof of (5.21).

Construction (5.22)

We will construct the projective limit of the models $\mathfrak{M}_n$. Thereunto we need to define mappings $\kappa_n : \mathfrak{M}_{n+1} \longrightarrow \mathfrak{M}_n$, for all $n \geq 1$ as follows. First we will define

$$\kappa_n : A_{n+1} \longrightarrow A_n.$$

Let $[p]_{n+1} \in A_{n+1}$, then we define $\kappa_n\big([p]_{n+1}\big) = [p]_n$. Now we will define

$$\kappa_n : H_{n+1} \longrightarrow H_n.$$

Let f be an element of H_{n+1}. We define

$$\kappa_n(f) : A_n \longrightarrow A_n$$

to be the following map. Let $[p]_n$ be in A_n, then $\kappa_n(f)\big([p]_n\big) = \kappa_n \circ f\big(\big[\pi'_n(p)\big]_{n+1}\big)$. It is obvious that $\kappa_n(f)$ is well-defined. So let us verify that $\kappa_n(f)$ is an element of H_n. First we are to show that $\kappa_n(f)$ is an laminal function. It is evident that $\kappa_n\big([\delta]_n\big) = [\delta]_n$. To verify that $\kappa_n(f)$ distributes over the alternative composition, we will need the fact that κ_n distributes over the alternative composition. Both proofs are trivial. The following remains to be shown. Let $[p]_n$ be in A_n. Suppose that $f\big(\big[\pi'_n(p)\big]_{n+1}\big) = [q]_{n+1}$. Then it is clear that $\kappa_n(f)\big([p]_n\big) = [q]_n$. We will prove that for all $1 \leq l \leq n$ we have:

$$\kappa_n(f)\big(\big[\pi'_l(p)\big]_n\big) = \big[\pi'_l(q)\big]_n.$$

Consider the calculation hereinafter.

$$\begin{aligned}
\kappa_n(f)\big(\big[\pi'_l(p)\big]_n\big) &= \kappa_n \circ f\big(\big[\pi'_n \circ \pi'_l(p)\big]_{n+1}\big)\\
&= \kappa_n \circ f\big(\big[\pi'_l \circ \pi'_n(p)\big]_{n+1}\big)\\
&= \kappa_n\big(\big[\pi'_l(q)\big]_{n+1}\big) && \text{(since } f \in H_{n+1}\text{)}\\
&= \big[\pi'_l(q)\big]_n.
\end{aligned}$$

So we find that $\kappa_n(f) \in H_n$. Now we will show that $\kappa_n : \mathfrak{M}_{n+1} \longrightarrow \mathfrak{M}_n$ is a homomorphism, that is, it distributes over all the operations. Recall that there are seven operations: the merge, the left-merge, the communication-merge, the alternative composition, the sequential composition, the composition of functions and the application function. Let $\star$ be one of $\|, \mathbin{\|\!\lfloor}, |, +$ or $\cdot$. Then we are to show that $\kappa_n(x \star y) = \kappa_n(x) \star \kappa_n(y)$, for all $x, y \in A_{n+1}$. This is trivial. We will show that κ_n distributes over the composition of functions. Let $f, g \in H_{n+1}$. Let $[p]_n$ be in A_n. Let $g\big(\big[\pi'_n(p)\big]_{n+1}\big) = [r]_{n+1}$ and let $f\big([r]_{n+1}\big) = [q]_{n+1}$. Now contemplate the following.

$$\begin{aligned}
\kappa_n(f \circ g)\big([p]_n\big) &= \kappa_n \circ (f \circ g)\big(\big[\pi'_n(p)\big]_{n+1}\big)\\
&= \kappa_n\big(f\big([r]_{n+1}\big)\big)\\
&= [q]_n\\
&= \kappa_n\big(\big[\pi'_n(q)\big]_{n+1}\big)\\
&= \kappa_n\big(f\big(\big[\pi'_n(r)\big]_{n+1}\big)\big)\\
&= \kappa_n(f)\big([r]_n\big)\\
&= \kappa_n(f)\big(\kappa_n\big([r]_{n+1}\big)\big)\\
&= \kappa_n(f)\Big(\kappa_n\big(g\big(\big[\pi'_n(p)\big]_{n+1}\big)\big)\Big)\\
&= \kappa_n(f) \circ \kappa_n(g)\big([p]_n\big).
\end{aligned}$$

Now we will show that κ_n distributes over the application function χ. Let $f \in H_{n+1}$ and let $[p]_{n+1}$ be in A_{n+1}. Suppose that $f\big([p]_{n+1}\big) = [q]_{n+1}$. Consider the following:

$$\begin{aligned}
\kappa_n\big(\chi(f, [p]_{n+1})\big) &= \kappa_n\big([q]_{n+1}\big)\\
&= [q]_n\\
&= \kappa_n\big(\big[\pi'_n(q)\big]_{n+1}\big)\\
&= \kappa_n \circ f\big(\big[\pi'_n(p)\big]_{n+1}\big)\\
&= \kappa_n(f)\big([p]_n\big)\\
&= \chi\big(\kappa_n(f), \kappa_n\big([p]_{n+1}\big)\big).
\end{aligned}$$

We constructed thus a row

$$\mathfrak{M}_1 \xleftarrow{\kappa_1} \mathfrak{M}_2 \xleftarrow{\kappa_2} \mathfrak{M}_3 \xleftarrow{\kappa_3} \mathfrak{M}_4 \longleftarrow \ldots \tag{11}$$

of models with homomorphisms between them. With this row, we will construct the inverse limit $\mathfrak{M}^\infty$. Let A^∞ and H^∞ be as follows:

$$\begin{aligned}
A^\infty &= \big\{\, (x_n)_n \;\big|\; x_n \in A_n, \kappa_n(x_{n+1}) = x_n \,\big\},\\
H^\infty &= \big\{\, (f_n)_n \;\big|\; f_n \in H_n, \kappa_n(f_{n+1}) = f_n \,\big\}.
\end{aligned}$$

We will call the elements of these sets projective rows. We will define the operations hereinafter. Let $\star$ be one of $\|$, $\lfloor\!\lfloor$, $|$, $+$ or $\cdot$. Let $(x_n)_n, (y_n)_n \in A^\infty$. Then we define $\star$ to be

$$(x_n)_n \star (y_n)_n = (x_n \star y_n)_n.$$

It is obvious that $(x_n \star y_n)_n \in A^\infty$. Now we will define the composition of functions. Let $(f_n)_n, (g_n)_n \in H^\infty$. Then we define

$$(f_n)_n \circ (g_n)_n = (f_n \circ g_n)_n.$$

From the fact that κ_n is a homomorphism it follows immediately that $(f_n \circ g_n)_n \in H^\infty$. Finally, we define the application function. Let $(f_n)_n \in H^\infty$ and let $(x_n)_n$ be in A^∞. Then we define χ to be

$$\chi\big((f_n)_n, (x_n)_n\big) = \big(\chi(f_n, x_n)\big)_n.$$

It will be clear that $\big(\chi(f_n, x_n)\big)_n \in A^\infty$. Now we will define constants of sort H^∞. Firstly we will introduce the encapsulation operator. To prevent any confusion we will label the encapsulation operators in all sets H_n as follows: $\partial^n_H \in H_n$. We define the encapsulation operator to be $\partial_H = (\partial^n_H)_n$. We will illustrate that $\partial_H \in H^\infty$. For every component of the encapsulation operator, we have of course $\partial^n_H \in H_n$. It remains thus to prove that ∂_H is a projective row. Let thereunto $[p]_n$ be in A_n. We easily deduce the following:

$$\begin{aligned}
\kappa_n(\partial^{n+1}_H)\big([p]_n\big) &= \kappa_n \circ \partial^{n+1}_H\big(\big[\pi'_n(p)\big]_{n+1}\big)\\
&= \big[\partial'_H \circ \pi'_n(p)\big]_n\\
&= \partial^n_H\big([p]_n\big).
\end{aligned}$$

So we find that $\kappa_n(\partial^{n+1}_H) = \partial^n_H$. Secondly we will introduce for all $k \geq 1$, the projection operators. We will label the projections in the sets H_n just as above: $\pi^n_k \in H_n$. Now we define the projection operator to be $\pi_k = (\pi^n_k)_n$. We will show that this projection is an element of H^∞. The first condition is satisfied by definition, so let us verify that $(\pi^n_k)_n$ is a projective row. Let $[p]_n$ be in A_n. Then it is easy to see the following:

$$\begin{aligned}
\kappa_n(\pi^{n+1}_k)\big([p]_n\big) &= \big[\pi'_k \circ \pi'_n(p)\big]_n\\
&= \pi^n_k\big([p]_n\big).
\end{aligned}$$

And we find $\kappa_n(\pi_k^{n+1}) = \pi_k^n$. At this point, we will introduce constants in A^∞. Let a be an atomic action. Then we define a row $\big([a]_n\big)_n$. We see immediately, by definition, that this row is in A^∞. Now let $\mathfrak{M}^\infty$ be the algebra that consists of the sets A^∞ and H^∞, the operators $+, \cdot, \|, \lfloor\!\lfloor, |, \chi$ and $\circ$, and the constants $\big([a]_n\big)_n \in A^\infty$ for all $a \in A$ and the constants $\pi_k, \partial_H \in H^\infty$, for all $k \geq 1$ and $H \subseteq A$. We will define the projections

$$\zeta_n : \mathfrak{M}^\infty \longrightarrow \mathfrak{M}_n$$

as follows. $\zeta_n(x) = x_n$, and $\zeta(f) = f_n$ if $x = (x_n)_n$ and $f = (f_n)_n$. It is very easy to see that for all $n \geq 1$, we have $\kappa_n \circ \zeta_{n+1} = \zeta_n$. It is a well-known fact that such a construction forms the projective or inverse limit. We find thus that $\mathfrak{M}^\infty$ is the projective limit of the row in expression (11).

Theorem (5.23)

$$\mathfrak{M}^\infty \models ACP_u.$$

See table 5 on page 39 for the axioms of ACP_u.

Proof. We know from theorem (5.10) that for all $n \geq 1 : \mathfrak{M}_n \models ACP_u$. Forasmuch as the operations on $\mathfrak{M}^\infty$ are defined component by component, we see at once that $\mathfrak{M}^\infty \models ACP_u$. This will end the proof of (5.23).

Theorem (5.24)

$$\mathfrak{M}^\infty \models RDP, RSP.$$

Proof. We will use the notations that we have already introduced in theorem (5.17). So let

$$V = \{x_\alpha : \alpha \in I\}$$

be a set of variables and let

$$E(V) = \big\{\, x_\alpha = t_\alpha\big((x_\beta)_{\beta\in I}\big) : \alpha \in I \,\big\}$$

be a guarded recursive specification. Without loss of generality, we may assume that $E(V)$ is completely guarded; see definition (2.15). For all $n \geq 1$ we have constructed in theorem (5.17) a solution $([p^n_\alpha])_\alpha$ for $E(V)$ in $\mathfrak{M}_n$. We claim that $\big([p^n_\alpha]_n\big)_n$ is an element of A^∞ for all $\alpha \in I$. This follows immediately from the following:

$$\begin{aligned}
\kappa_n\big(\big[p^{n+1}_\alpha\big]_{n+1}\big) &= \big[p^{n+1}_\alpha\big]_n \\
&= \big[\pi'_n(p^{n+1}_\alpha)\big]_n \\
&= \big[\pi'_n(p^{n}_\alpha)\big]_n && \text{see (3)} \\
&= [p^n_\alpha]_n.
\end{aligned}$$

We are to verify that for all $\alpha \in I : \big([p^n_\alpha]_n\big)_n = \big(\big[t_\alpha(p^n_\beta)_\beta\big]_n\big)_n$. But we know from the proof of theorem (5.17) that for all $n \geq 1$ we have $[p^n_\alpha]_n = \big[t_\alpha(p^n_\beta)_\beta\big]_n$; so we see that $\mathfrak{M}^\infty \models RDP$, too. Now suppose that for all $\alpha \in I$ we have $\big([q^n_\alpha]_n\big)_n = \big(\big[t_\alpha(q^n_\beta)_\beta\big]_n\big)_n$. Let $n \geq 1$ be fixed. Since RSP is valid in $\mathfrak{M}_n$, we see that $[q^n_\alpha]_n = [p^n_\alpha]_n$, so we also find $\big([p^n_\alpha]_n\big)_n = \big([q^n_\alpha]_n\big)_n$ and $\mathfrak{M}^\infty \models RSP$. This ends the proof of theorem (5.24).

Theorem (5.25)

$$\mathfrak{M}^\infty \models AIP.$$

Proof. Let $(x_n)_n, (y_n)_n \in A^\infty$. Suppose that for all $k \geq 1$, we have $\pi_k\big((x_n)_n\big) = \pi_k\big((y_n)_n\big)$. Then we find for all $n, k \geq 1$

$$\pi_k(x_n) = \pi_k(y_n).$$

Now fix an $n \geq 1$; we know that $\mathfrak{M}_n \models AIP$, so we find $x_n = y_n$. But this is valid for all $n \geq 1$ and we find thus $(x_n)_n = (y_n)_n$. This is precisely what we wanted to prove.

Theorem (5.26)

$$\mathfrak{M}^\infty \models EA.$$

Proof. Since H^∞ is a function space, we immediately have this property. See also the proof of (5.20).

Theorem (5.27)

$$\mathfrak{M}^\infty \models ODP, OSP.$$

Proof. We will modify the notations of theorem (5.21) slightly. In there we defined a sequence of elements

$$\mu_1^r, \ldots, \mu_l^r \in H_n$$

for a fixed $n \geq 1$ and $1 \leq r \leq n$. We will give this sequence of elements an extra label as follows:

$$\mu_1^{n,r}, \ldots, \mu_l^{n,r} \in H_n.$$

We define for all i with $1 \leq i \leq l$ the row μ_i to be $(\mu_i^{n,n})_n$. We claim that

$$\mu_1, \ldots, \mu_l \in H^\infty. \tag{12}$$

To prove (12) we fix an i with $1 \leq i \leq l$. By definition, we see that for all $n \geq 1$ we have $\mu_i^{n,n} \in H_n$, so we have to verify that μ_i is a projective row, that is, we are to show that for all $n \geq 1$

$$\kappa_n(\mu_i^{n+1,n+1}) = \mu_i^{n,n}. \tag{13}$$

First, we will treat the case $n = 1$. Let $[p]_1$ be as in equation (6) of theorem (5.21). It is very easy to see that

$$\begin{aligned}\kappa_1(\mu_1^{2,2})\big([p]_1\big) &= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_1 + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_1\\ &= \mu_1^{1,1}\big([p]_1\big).\end{aligned}$$

Now let henceforward $n > 1$. In order to verify (13) we will need the following intermediate result. For all $1 \leq i \leq l$ and for all $1 \leq r \leq n$, we have

$$\kappa_n(\mu_i^{n+1,r}) = \mu_i^{n,r}. \tag{14}$$

Let $r = 1$ and let $1 \leq i \leq l$ be chosen. Let $[p]_n \in A_n$ be as in equation (6). Consider the following.

$$\begin{aligned}\kappa_n(\mu_i^{n+1,1})\big([p]_n\big) &= \kappa_n \circ \mu_i^{n+1,1}\big(\big[\pi_n'(p)\big]_{n+1}\big)\\ &= \sum_{j=1}^{s}\big[a_j^{(i)}\big]_n + \sum_{k=1}^{t}\big[b_k^{(i)}\big]_n\\ &= \mu_1^{n,1}\big([p]_n\big).\end{aligned}$$

Now let $1 \le r < n$ and suppose that (14) is proved up to and including r. We will prove it for $r+1$.

$$\begin{aligned}
\kappa_n(\mu_i^{n+1,r+1})([p]_n) &= \kappa_n \circ \mu_i^{n+1,r+1}\big([\pi'_n(p)]_{n+1}\big) \\
&= \sum_{j=1}^{s} [a_j^{(i)}]_n \cdot \kappa_n(\mu_{\sigma(a_j,i)}^{n+1,r})\big([\pi'_{n-1}(p_j)]_n\big) + \sum_{k=1}^{t} [b_k^{(i)}]_n \\
&= \sum_{j=1}^{s} [a_j^{(i)}]_n \cdot \mu_{\sigma(a_j,i)}^{n,r}\big([\pi'_{n-1}(p_j)]_n\big) + \sum_{k=1}^{t} [b_k^{(i)}]_n \\
&= \mu_i^{n,r+1}\big([\pi'_n(p)]_n\big) \\
&= \mu_i^{n,r+1}\big([p]_n\big).
\end{aligned}$$

This ends the proof of (14). In particular, we find for $r = n$

$$\kappa_n(\mu_i^{n+1,n}) = \mu_i^{n,n}. \tag{15}$$

Now we will verify that equation (13) is valid. Let $[p]_n$ still be as in equation (6). We will calculate the left-hand side of equation (13).

$$\begin{aligned}
\kappa_n(\mu_i^{n+1,n+1})([p]_n) &= \sum_{j=1}^{s} [a_j^{(i)}]_n \cdot \kappa_n(\mu_{\sigma(a_j,i)}^{n+1,n})\big([\pi'_{n-1}(p_j)]_n\big) + \sum_{k=1}^{t} [b_k^{(i)}]_n \\
&= \sum_{j=1}^{s} [a_j^{(i)}]_n \cdot \mu_{\sigma(a_j,i)}^{n,n}\big([\pi'_{n-1}(p_j)]_n\big) + \sum_{k=1}^{t} [b_k^{(i)}]_n. \qquad \text{with (15)}
\end{aligned}$$

We know that all $\mu_i^{n,n}$ satisfy equations (4) and (5) of theorem (5.21). So we find for the right-hand side of equation (13):

$$\begin{aligned}
\mu_i^{n,n}\big([p]_n\big) &= \mu_i^{n,n}\big([\pi'_n(p)]_n\big) \\
&= \sum_{j=1}^{s} [a_j^{(i)}]_n \cdot \mu_{\sigma(a_j,i)}^{n,n}\big([\pi'_{n-1}(p_j)]_n\big) + \sum_{k=1}^{t} [b_k^{(i)}]_n,
\end{aligned}$$

and therewith we deduced that $\mu_i \in H^\infty$. In order to prove the existential part of the theorem, i.e., $\mathfrak{M}^\infty \models ODP$, it suffices to show that for $\mu_1, \ldots, \mu_l$ the following two equations hold.

$$\begin{aligned}
\mu_i\big(([a]_n)_n\big) &= \big([a^{(i)}]_n\big)_n, \\
\mu_i\big(([a]_n)_n \cdot (x_n)_n\big) &= \big([a^{(i)}]_n\big)_n \cdot \mu_{\sigma(a,i)}\big((x_n)_n\big),
\end{aligned}$$

in which $a \in A$ and $(x_n)_n \in H^\infty$. This is trivial. We find thus that $\mathfrak{M}^\infty \models ODP$. Now we will prove the uniqueness part. Suppose that $\nu_1, \ldots, \nu_l \in H^\infty$ solves the system $E(M)$ in $\mathfrak{M}^\infty$, too. With $\nu_i = (\nu_i^n)_n$. Let $p \in G$. It is easy to infer that for all $n \ge 1$ we have

$$\begin{aligned}
\nu_i^n\big([a]_n\big) &= [a^{(i)}]_n, \\
\nu_i^n\big([a]_n \cdot [p]_n\big) &= [a^{(i)}]_n \cdot \nu_{\sigma(a,i)}^n\big([p]_n\big).
\end{aligned}$$

But because of theorem (5.21), we know that $\nu_i^n = \mu_i^{n,n}$. So we find $\nu_i = (\nu_i^n)_n = (\mu_i^{n,n})_n = \mu_i$. So $\mathfrak{M}^\infty \models OSP$. This will finish the proof of (5.27).

6. Applications

In this section we will apply the theory $ACP_{\tau,u}$ on the one hand, by proving a number of basic adversaria in a "new" way. By this we mean an enumeration of straightforward non-related results (at first sight) that are already known. The main difference here is that we will use $ACP_{\tau,u}$ to prove these theorems. In fact, the correlation between these results is the use of auxiliary linear unary operators in order to prove them. On the other hand, it is not only customary to use auxiliary linear unary operators in verifications, but also in specifications. So we will give some examples of specifications with the aid of such auxiliary operators, too, in which we will use *ODP* and *OSP* to introduce the operators needed for the specification. In practice it is a combination of both: specification with the aid of linear unary operators in order to be able to give a verification.

The first example in using an auxiliary operator is a proof that $KFAR_1 \implies KFAR_2$. This theorem can be found in [**17**]. In this example, we will use linear unary operators in the body of the proof. But first we have to know the meaning of $KFAR_i$ for $i = 1, 2$. Therefore, we will need the definition below. This definition is taken from [**10**].

Principle (6.1) *Koomen's Fair Abstraction Rule*

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ be in P. Let $I \subseteq A$ and suppose that we have the following identities for these processes:

$$\begin{aligned} x_1 &= i_1 \cdot x_2 + y_1, \\ x_2 &= i_2 \cdot x_3 + y_2, \\ &\vdots \\ x_{n-1} &= i_{n-1} \cdot x_n + y_{n-1}, \\ x_n &= i_n \cdot x_1 + y_n, \end{aligned}$$

with the following assumptions on the i_j, $(1 \leq j \leq n)$: $\{\tau\} \neq \{i_1, \ldots, i_n\} \subseteq I \cup \{\tau\}$, then we have:

$$\tau_I(x) = \tau \cdot \big(\tau_I(y_1) + \tau_I(y_2) + \cdots + \tau_I(y_n)\big).$$

We will refer to this principle with the abbreviation $KFAR_n$.

Theorem (6.2)

Suppose that $KFAR_1$ holds, then we can derive that $KFAR_2$ holds.

Proof. We will prove this theorem only in the following case. Let u and v be closed terms. Let $i \in I$ and $j \in I$ be internal atomic actions ($I \subseteq A$). Let x and y be processes such that the following holds:

$$\begin{aligned} x &= i \cdot y + u, \\ y &= j \cdot x + v. \end{aligned}$$

We are to show:

$$\tau_I(x) = \tau \cdot \tau_I(u + v).$$

We will use the following abbreviations: $u' = \tau_{\{i\}}(u)$ and $v' = \tau_{\{i\}}(v)$. Consider the guarded recursive specification E_1 below:

$$E_1 = \big\{\, x_1 = \tau \cdot y_1 + u',\, y_1 = j \cdot x_1 + v' \,\big\}.$$

(At this point we already see why we consider the closed term case only: the specification E_1 must be without the abstraction operator. Although u' and v' are abbreviations in which the abstraction operator occurs, they are closed terms, so we can eliminate the $\tau_{\{i\}}$ with the axioms concerning the abstraction operator. Hence, we should consider a completely equivalent guarded recursive specification E_1' without the abstraction operator. We will not do so. If we want to prove the general case, we will probably need the notion of guarded recursive specifications with parameters. Then the closed terms u and v can be interpreted as auxiliary processes that are parameters of a guarded recursive specification. The processes u' and v' are in that case just other parameters, hence, we still have a guarded recursive specification without the abstraction operator. The concept of guarded recursive specifications with parameters, can be found in [**15**].)

It is very easy to see that the conditions of theorem (4.12) are satisfied, so we see that $\tau_{\{i\}}$ is idempotent. This can be used to see that the unique solution of E_1 is $\big(\tau_{\{i\}}(x), \tau_{\{i\}}(y)\big)$. We will use the fact that $\tau_{\{i\}}$ is idempotent tacitly in the sequel. Now consider the guarded recursive specification E_2:

$$E_2 = \big\{\, x_2 = i \cdot y_2 + u' + v',\, y_2 = j \cdot x_2 + v' \,\big\}.$$

Let (x_2, y_2) be the solution of E_2. Then it is clear that we have the following for $\tau_{\{i\}}(y_2)$ and $\tau_{\{i\}}(x_2)$:

$$\tau_{\{i\}}(y_2) = j \cdot \tau_{\{i\}}(x_2) + v'$$

and

$$\begin{aligned}
\tau_{\{i\}}(x_2) &= \tau \cdot \tau_{\{i\}}(y_2) + u' + v' \\
&= \tau \cdot \tau_{\{i\}}(y_2) + \tau_{\{i\}}(y_2) + u' + v' \\
&= \tau \cdot \tau_{\{i\}}(y_2) + j \cdot \tau_{\{i\}}(x_2) + v' + v' + u' \\
&= \tau \cdot \tau_{\{i\}}(y_2) + j \cdot \tau_{\{i\}}(x_2) + v' + u' \\
&= \tau \cdot \tau_{\{i\}}(y_2) + \tau_{\{i\}}(y_2) + u' \\
&= \tau \cdot \tau_{\{i\}}(y_2) + u'.
\end{aligned}$$

Hence, $\big(\tau_{\{i\}}(x_2), \tau_{\{i\}}(y_2)\big)$ is also a solution for the guarded recursive specification E_1, so by *RSP* we obtain: $\tau_{\{i\}}(x_2) = \tau_{\{i\}}(x)$. With the aid of (4.18), it is easy to see that τ_I is a right-absorber for $\tau_{\{i\}}$, that is, we know that $\tau_I \circ \tau_{\{i\}} = \tau_I$. Thus, we achieve:

$$\begin{aligned}
\tau_I(x_2) &= \tau_I \circ \tau_{\{i\}}(x_2) \\
&= \tau_I \circ \tau_{\{i\}}(x) \\
&= \tau_I(x).
\end{aligned} \tag{1}$$

Now consider the guarded recursive specifications E_3 and E_4:

$$\begin{aligned}
E_3 &= \big\{\, x_3 = i \cdot y_3 + u'' + v'',\, y_3 = \tau \cdot x_3 + u'' + v'' \,\big\}, \\
E_4 &= \big\{\, x_4 = i \cdot y_4 + u' + v',\, y_4 = j \cdot x_4 + u' + v' \,\big\}.
\end{aligned}$$

We used above the abbreviations $\tau_{\{i,j\}}(u) = u''$ and $\tau_{\{i,j\}}(v) = v''$. With the aid of corollary (4.22) it is is immediately clear that $\tau_{\{j\}} \circ \tau_{\{i\}} = \tau_{\{i,j\}}$. Now we can derive:

$$\tau_{\{j\}}(x_2) = i \cdot \tau_{\{j\}}(y_2) + u'' + v''$$

and

$$\begin{aligned}
\tau_{\{j\}}(y_2) &= \tau \cdot \tau_{\{j\}}(x_2) + v'' \\
&= \tau \cdot \tau_{\{j\}}(x_2) + \tau_{\{j\}}(x_2) + v'' \\
&= \tau \cdot \tau_{\{j\}}(x_2) + i \cdot \tau_{\{j\}}(y_2) + u'' + v'' + v'' \\
&= \tau \cdot \tau_{\{j\}}(x_2) + i \cdot \tau_{\{j\}}(y_2) + u'' + v'' + u'' + v'' \\
&= \tau \cdot \tau_{\{j\}}(x_2) + \tau_{\{j\}}(x_2) + u'' + v'' \\
&= \tau \cdot \tau_{\{j\}}(x_2) + u'' + v''.
\end{aligned}$$

We see that $\big(\tau_{\{j\}}(x_2), \tau_{\{j\}}(y_2)\big)$ is a solution for E_3. For the solution (x_4, y_4) of E_4 we can deduce easily:

$$\tau_{\{j\}}(x_4) = i \cdot \tau_{\{j\}}(y_4) + u'' + v''$$

and

$$\tau_{\{j\}}(y_4) = \tau \cdot \tau_{\{j\}}(x_4) + u'' + v''.$$

Now we see that $\big(\tau_{\{j\}}(x_4), \tau_{\{j\}}(y_4)\big)$ is also a solution for E_3. Hence, by *RSP* we obtain $\tau_{\{j\}}(x_4) = \tau_{\{j\}}(x_2)$. Completely analogous to the calculation of equation (1), we find:

$$\tau_I(x_4) = \tau_I(x_2). \tag{2}$$

Observe that we use here (4.18) since we use that τ_I is a right-absorber for $\tau_{\{j\}}$. Consider the guarded recursive specifications E_5 and E_6:

$$\begin{aligned} E_5 &= \big\{\, x_5 = j \cdot x_5 + u' + v' \,\big\}, \\ E_6 &= \big\{\, x_6 = j \cdot y_6 + u' + v',\ y_6 = j \cdot x_6 + u' + v' \,\big\}. \end{aligned}$$

At this point we want to introduce a linear unary operator. Let n be a function name. Consider the linear functional specification below:

$$\begin{aligned} E(n) = \big\{\, n(a) = a \bigm| a \in A \setminus \{i\} \,\big\} &\cup \{n(i) = j\} \\ &\cup \big\{\, n(a \cdot x) = n(a) \cdot n(x) \mid a \in A \,\big\}. \end{aligned}$$

With the aid of *ODP* we know that there is a valuation φ which solves this specification. Let us say $\varphi(n) = \rho \in F$. With the aid of theorem (4.15), we find that $\tau_{\{i\}}$ is a left-absorber for ρ. So for the solution (x_4, y_4) of E_4, we can derive:

$$\begin{aligned} \rho(x_4) &= j \cdot \rho(y_4) + u' + v', \\ \rho(y_4) &= j \cdot \rho(x_4) + u' + v'. \end{aligned}$$

So we see that $\big(\rho(x_4), \rho(y_4)\big)$ is a solution for E_6. On the other hand, if we interchange the order of the equations above, we see that $\big(\rho(y_4), \rho(x_4)\big)$ is a solution for E_6, thus, with the aid of *RSP*, we obtain $\rho(x_4) = \rho(y_4)$. In particular we find:

$$\rho(x_4) = j \cdot \rho(x_4) + u' + v'.$$

Hence, $\rho(x_4)$ is a solution for E_5. Let x_5 be the solution for E_5, then we obtain by *RSP* $\rho(x_4) = x_5$. We will use theorem (4.18) to see that τ_I is a right-absorber for ρ. We will use this fact in the calculation hereinafter. On the one hand we have:

$$\begin{aligned} \tau_I(x_5) &= \tau_I \circ \rho(x_4) \\ &= \tau_I(x_4) \\ &= \tau_I(x_2) && \text{because of (2)} \\ &= \tau_I(x). && \text{because of (1)} \end{aligned}$$

And on the other hand we may apply $KFAR_1$ to the equation $x_5 = j \cdot x_5 + u' + v'$. This gives the following: (notice that we use the fact that τ_I is a right-absorber for $\tau_{\{i\}}$)

$$\begin{aligned} \tau_I(x_5) &= \tau \cdot \tau_I(u' + v') \\ &= \tau \cdot \big(\tau_I \circ \tau_{\{i\}}(u) + \tau_I \circ \tau_{\{i\}}(v)\big) \\ &= \tau \cdot \big(\tau_I(u) + \tau_I(v)\big) \\ &= \tau \cdot \tau_I(u + v). \end{aligned}$$

Combining these two inferences, we conclude the proof of (6.2).

In our next example, we will give a recursive specification of a queue Q over a data set D with input channel 1 and output channel 2. We suppose that the data set D contains more than one datum (see figure 3).

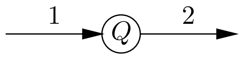

Figure 3. A queue Q with input channel 1 and output channel 2.

It is known that an infinite guarded recursive specification of Q can be given by the equations hereinafter:

$$Q = Q_\lambda = \sum_{d \in D} r_1(d) \cdot Q_d, \tag{3}$$

$$Q_{\sigma * d} = s_2(d) \cdot Q_\sigma + \sum_{e \in D} r_1(e) \cdot Q_{e * \sigma * d}, \tag{4}$$

for any word $\sigma \in D^*$ and any $d \in D$. Here, we use $\lambda \in D^*$ for the empty word. The asterisk $(*)$ stands for the concatenation of words. It is known that there is no finite guarded recursive specification over *ACP* which defines the process Q. It is also known that the queue can be specified with the aid of a finite guarded recursive specification, if we allow certain auxiliary linear unary operators (renaming operators). Both results can be found in [**2**]. Below we will state the latter theorem in terms of $ACP_{\tau,u}$.

Theorem (6.3)

The queue is definable by a finite guarded recursive specification in $ACP_{\tau,u}$.

Proof. Suppose that we have for the set A of atomic actions

$$\{r_1(d), s_2(d), l(d), u(d) : d \in D\} \subseteq A.$$

Let the communication function be given as follows:

$$\forall\, d \in D : l(d) \mid l(d) = u(d),$$

and all the other communications result in δ. Consider the following linear functional specification for the set of function names $N = \{n, m\}$:

$$\begin{aligned} E(N) = {} & \big\{\, n\big(u(d)\big) = s_2(d) : d \in D \,\big\} \cup \big\{\, n\big(l(d)\big) = \delta : d \in D \,\big\} \\ & \cup \big\{\, n(a) = a : a \in A \setminus \big(u(D) \cup l(D)\big) \,\big\} \\ & \cup \big\{\, m\big(s_2(d)\big) = l(d) : d \in D \,\big\} \cup \big\{\, m(a) = a : a \in A \setminus s_2(D) \,\big\} \\ & \cup \big\{\, f(a \cdot x) = f(a) \cdot f(x) \bigm| f \in N, a \in A \,\big\}. \end{aligned}$$

We used above the following abbreviations: $u(D) = \{u(d) : d \in D\}$, $l(D) = \{l(d) : d \in D\}$ and $s_2(D) = \{s_2(d) : d \in D\}$. According to *ODP*, there is a valuation $\varphi : N \longrightarrow F$. Let us say $\varphi(n) = \nu \in F$ and $\varphi(m) = \mu \in F$. Consider the following finite guarded recursive specification:

$$\begin{aligned} R &= \sum_{d \in D} r_1(d) \cdot \nu\big(\mu(R) \parallel s_2(d) \cdot Z\big), \\ Z &= \sum_{d \in D} l(d) \cdot Z. \end{aligned}$$

Now let $\sigma \in D^*$, then we define the process R_σ inductively as displayed below:

$$\begin{aligned} R_\lambda &= R, \\ R_{\sigma * d} &= \nu\big(\mu(R_\sigma) \parallel s_2(d) \cdot Z\big). \end{aligned}$$

At this point, we will prove the following claim:

$$R_\sigma = \nu\big(\mu(R_\sigma) \parallel Z\big). \tag{5}$$

We will verify this by showing that both left and right-hand side of equation (5) are solutions for the same guarded recursive specification and then we will apply *RSP*. In order to prove this, we need another intermediate result:

$$\forall\, \sigma \in D^*, \forall\, d \in D: \quad \nu\big(\mu(R_\sigma) \mathbin{\|\!\_} s_2(d) \cdot Z\big) = \sum_{e \in D} r_1(e) \cdot R_{e*\sigma*d}. \tag{6}$$

We will prove (6) with induction to the length of the word σ. First let $d \in D$ be arbitrarily chosen and let $\sigma = \lambda$. Consider the following calculation:

$$\begin{aligned}
\nu\big(\mu(R_\lambda) \mathbin{\|\!_} s_2(d) \cdot Z\big) &= \nu\Big(\sum_{e \in D} r_1(e) \cdot \mu(R_e) \mathbin{\|\!_} s_2(d) \cdot Z\Big) \\
&= \sum_{e \in D} r_1(e) \cdot \nu\big(\mu(R_{e*\lambda}) \parallel s_2(d) \cdot Z\big) \\
&= \sum_{e \in D} r_1(e) \cdot R_{e*\lambda*d}.
\end{aligned}$$

Hence, for $\sigma = \lambda$ equation (6) is proved. Now let $d \in D$ and let $\sigma = \rho * f \in D^*$ for some $f \in D$, such that equation (6) is proved for $\rho \in D^*$. We show that (6) holds for σ:

$$\begin{aligned}
\nu\big(\mu(R_\sigma) \mathbin{\|\!_} s_2(d) \cdot Z\big) &= \nu\Big(\mu \circ \nu\big(\mu(R_\rho) \parallel s_2(f) \cdot Z\big) \mathbin{\|\!_} s_2(d) \cdot Z\Big) \\
&= \nu\Big(\mu \circ \nu\big(\mu(R_\rho) \mathbin{\|\!_} s_2(f) \cdot Z\big) \mathbin{\|\!_} s_2(d) \cdot Z\Big) \\
&\qquad + \nu\Big(\mu \circ \nu\big(s_2(f) \cdot Z \mathbin{\|\!_} \mu(R_\rho)\big) \mathbin{\|\!_} s_2(d) \cdot Z\Big) \\
&= \nu\Big(\mu \circ \nu\big(\mu(R_\rho) \mathbin{\|\!_} s_2(f) \cdot Z\big) \mathbin{\|\!_} s_2(d) \cdot Z\Big) + \delta \\
&= \nu\Big(\mu\Big(\sum_{e \in D} r_1(e) \cdot R_{e*\rho*f}\Big) \mathbin{\|\!_} s_2(d) \cdot Z\Big) \\
&= \sum_{e \in D} r_1(e) \cdot \nu\big(\mu(R_{e*\sigma}) \parallel s_2(d) \cdot Z\big) \\
&= \sum_{e \in D} r_1(e) \cdot R_{e*\sigma*d}.
\end{aligned}$$

This ends the proof of our intermediate result stated in (6). Let us now calculate the left-hand side of equation (5), first we take $\sigma = \lambda$:

$$\begin{aligned}
R_\lambda &= R \\
&= \sum_{d \in D} r_1(d) \cdot \nu\big(\mu(R_\lambda) \parallel s_2(d) \cdot Z\big) \\
&= \sum_{d \in D} r_1(d) \cdot \nu\big(\mu(R) \parallel s_2(d) \cdot Z\big) \\
&= \sum_{d \in D} r_1(d) \cdot R_d.
\end{aligned}$$

Now we will drop the assumption of σ being the empty word.

$$\begin{aligned}
R_{\sigma*d} &= \nu\big(\mu(R_\sigma) \parallel s_2(d)\cdot Z\big)\\
&= \nu\big(\mu(R_\sigma) \mathbin{|\!\!\lfloor} s_2(d)\cdot Z\big) + s_2(d)\cdot \nu\big(\mu(R_\sigma) \parallel Z\big)\\
&= \sum_{e\in D} r_1(e)\cdot R_{e*\sigma*d} + s_2(d)\cdot \nu\big(\mu(R_\sigma) \parallel Z\big).
\end{aligned} \tag{7}$$

Consider the guarded recursive specification G below:

$$G \;=\; \Big\{ X_\lambda = \sum_{d\in D} r_1(d)\cdot X_d,$$
$$X_{\sigma*d} = \sum_{e\in D} r_1(e)\cdot X_{e*\sigma*d} + \nu\big(\mu(X_\sigma) \parallel Z\big) \;\Big|\; d\in D, \sigma\in D^* \Big\}.$$

Then it is obvious that, if we put $X_\sigma = R_\sigma$ for all $\sigma \in D^*$, this system is a solution for G. Observe that the linear unary operators, appearing in the guarded recursive specification G, are concrete operators; confer remark (2.19). Now we will prove that the right-hand side of equation (5) is also a solution for the specification G. Therefore, we will introduce one more abbreviation: $S_\sigma = \nu\big(\mu(R_\sigma) \parallel Z\big)$. Consider the calculation below for S_λ:

$$\begin{aligned}
S_\lambda &= \nu\big(\mu(R_\lambda) \mathbin{|\!\!\lfloor} Z\big)\\
&= \nu\Big(\mu\Big(\sum_{d\in D} r_1(d)\cdot \nu\big(\mu(R_\lambda) \parallel s_2(d)\cdot Z\big)\Big) \mathbin{|\!\!\lfloor} Z\Big)\\
&= \nu\Big(\sum_{d\in D} r_1(d)\cdot \mu\circ\nu\big(\mu(R_\lambda) \parallel s_2(d)\cdot Z\big) \mathbin{|\!\!\lfloor} Z\Big)\\
&= \sum_{d\in D} r_1(d)\cdot \nu\big(\mu\circ\nu\big(\mu(R_\lambda) \parallel s_2(d)\cdot Z\big) \parallel Z\big)\\
&= \sum_{d\in D} r_1(d)\cdot \nu\big(\mu(R_d) \parallel Z\big)\\
&= \sum_{d\in D} r_1(d)\cdot S_d.
\end{aligned}$$

Now we calculate $S_{\sigma*d}$:

$$\begin{aligned}
S_{\sigma*d} &= \nu\big(\mu(R_{\sigma*d}) \parallel Z\big)\\
&= \sum_{e\in D} r_1(e)\cdot \nu\big(\mu(R_{e*\sigma*d}) \parallel Z\big) + s_2(d)\cdot \nu\big(\mu(S_\sigma) \parallel Z\big) && \text{because of (7)}\\
&= \sum_{e\in D} r_1(e)\cdot S_{e*\sigma*d} + s_2(d)\cdot \nu\big(\mu(S_\sigma) \parallel Z\big)
\end{aligned}$$

It is clear that, if we put for all $\sigma \in D^* : X_\sigma = S_\sigma$, this system is also a solution for G. Hence, by *RSP*, we conclude the proof of equation (5). Now we find for the defining equations for R_σ:

$$\begin{aligned}
R_\lambda &= \sum_{d\in D} r_1(d)\cdot R_d,\\
R_{\sigma*d} &= \sum_{e\in D} r_1(d)\cdot R_{e*\sigma*d} + s_2(d)\cdot R_\sigma.
\end{aligned}$$

But these are the defining equations of the process $Q = Q_\lambda$; see equations (3) and (4). Thus, again using *RSP*, we obtain $Q = R_\lambda$. This ends the proof of (6.3).

At this point, we will give an example in which we use the operator definition principle in order to specify a communication network. In [**2**] this particular example is specified with an auxiliary operator, which is called the localization operator. In fact, this is the composition of a renaming operator and an abstraction operator. We will handle this in a slightly other way. We will combine three operators in one linear unary operator instead of two: a renaming operator (which is the auxiliary part), an abstraction operator and an encapsulation operator. In this network, we will have an environment E, a sender S and a receiver R. We will have four channels 1–4. We will depict this network in figure 4.

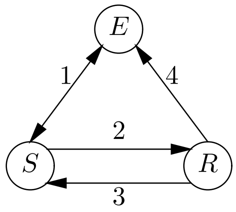

Figure 4. A communication network.

Let D be a finite set of data and let $ack \notin D$ be an acknowledgement. We will give the recursive equations for the sender S and the receiver R.

$$S = \sum_{d \in D} r_1(d) \cdot s_2(d) \cdot r_3(ack) \cdot s_1(ack) \cdot S,$$
$$R = \sum_{d \in D} r_2(d) \cdot s_4(d) \cdot s_3(ack) \cdot R.$$

The sender and the receiver both communicate with the environment via channels 1 and 4. The environment E can send data along 1, receive an ack along 1, or receive data along 4. Thus, we can put for the recursive specification for E:

$$E = \Big(\sum_{d \in D} s_1(d) + \sum_{d \in D} r_4(d) + r_1(ack) \Big) \cdot E.$$

The behaviour of the environment is that we first must have an $s_1(d)$, then an $r_1(d)$, then an $r_4(d)$ and then an $r_1(ack)$ before the next $s_1(e)$ can follow. This ordering of atomic actions is not expressed in the definition of E. We will use an auxiliary linear unary operator to express this behaviour. We assume for the set of atomic actions A the following:

$$\begin{aligned} A = {} & \{\, r_i(d), s_i(d), c_i(d) : d \in D,\ i = 1,2,4 \,\} \\ & \cup \{\, r_i(ack), s_i(ack), c_i(ack) : i = 1,3 \,\}. \end{aligned}$$

Next we will give, for a function name n, the following linear functional specification:

$$\begin{aligned} E(n) = {} & \{\, n\big(r_i(x)\big) = n\big(s_i(x)\big) = \delta \;\big|\; i = 1,2,4,\ x \in D \cup \{ack\} \} \\ & \cup \{\, n\big(c_1(x)\big) = s_1(x) \;\big|\; x \in D \cup \{ack\} \} \\ & \cup \{\, n\big(c_2(d)\big) = \tau \;\big|\; d \in D \,\} \\ & \cup \{\, n\big(c_4(d)\big) = r_4(d) \;\big|\; d \in D \,\} \\ & \cup \{ n\big(c_3(ack)\big) = \tau \} \\ & \cup \{\, n(a \cdot x) = n(a) \cdot n(x) \;\big|\; a \in A \,\}. \end{aligned}$$

With the aid of *ODP*, we know that there is a valuation φ that solves this system. Let us say $\varphi(n) = \nu \in F$ (notice that this linear unary operator is an example of an abstracting operator). We can now state the following theorem in which we will use the notations that we have introduced above. This theorem says that we will obtain the actions of E in the right order.

Theorem (6.4)

$$\nu(E \parallel S \parallel R) = \sum_{d \in D} s_1(d) \cdot r_4(d) \cdot r_1(ack) \cdot \nu(E \parallel S \parallel R).$$

Proof. This consists of the following straightforward calculation:

$$\begin{aligned}
\nu(E \parallel S \parallel R) &= \nu\big((E \mid S) \mathbin{⌊\!⌊} R\big) \\
&= \sum_{d \in D} s_1(d) \cdot \nu\big(s_2(d) \cdot r_3(ack) \cdot s_1(ack) \cdot S \parallel E \parallel R\big) \\
&= \sum_{d \in D} s_1(d) \cdot \nu\big(\big(s_2(d) \cdot r_3(ack) \cdot s_1(ack) \cdot S \mid R\big) \mathbin{⌊\!⌊} E\big) \\
&= \sum_{d \in D} s_1(d) \cdot \tau \cdot \nu\big(s_4(d) \cdot s_3(ack) \cdot R \parallel r_3(ack) \cdot s_1(ack) \cdot S \parallel E\big) \\
&= \sum_{d \in D} s_1(d) \cdot \nu\big(\big(s_4(d) \cdot s_3(ack) \cdot R \mid E\big) \mathbin{⌊\!⌊} r_3(ack) \cdot s_1(ack) \cdot S\big) \\
&= \sum_{d \in D} s_1(d) \cdot r_4(d) \cdot \nu\big(s_3(ack) \cdot R \parallel E \parallel r_3(ack) \cdot s_1(ack) \cdot S\big) \\
&= \sum_{d \in D} s_1(d) \cdot r_4(d) \cdot \nu\big(\big(s_3(ack) \cdot R \mid r_3(ack) \cdot s_1(ack) \cdot S\big) \mathbin{⌊\!⌊} E\big) \\
&= \sum_{d \in D} s_1(d) \cdot r_4(d) \cdot \tau \cdot \nu\big(R \parallel s_1(ack) \cdot S \parallel E\big) \\
&= \sum_{d \in D} s_1(d) \cdot r_4(d) \cdot \nu\big(\big(E \mid s_1(ack) \cdot S\big) \mathbin{⌊\!⌊} R\big) \\
&= \sum_{d \in D} s_1(d) \cdot r_4(d) \cdot s_1(ack) \cdot \nu(E \parallel S \parallel R).
\end{aligned}$$

In the next example, we will describe a process P, which can be in different states. Suppose we have a finite data set D consisting of two elements 0 and 1. We have two input channels, 1 and 2, and one output channel 3. The process P reads in an arbitrary order the data of the input channels 1 and 2 and sends along channel 3 the sum of these data modulo 2 (and then it starts all over again). We can think of P as a simulation of a(n) `xor`-port with the aid of process algebra. We will depict this process in figure 5.

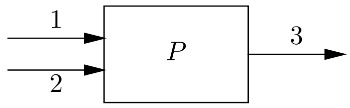

Figure 5. Simulation of a(n) `xor`-port.

We will first describe the process P without the merge. We assume that we have the following set of atomic actions:

$$\begin{aligned}
A &= \{r'_i(j) : i = 1, 2 \; j = 0, 1\} \cup \{t\} \\
&\quad \cup \{r_i(j) : i = 1, 2 \; j = 0, 1\} \cup \{s_3(j) : j = 0, 1\} \\
&= \{r'_i(j) : i = 1, 2 \; j = 0, 1\} \cup \{t\} \cup B,
\end{aligned}$$

with $B = \{r_i(j) : i = 1, 2\ j = 0, 1\} \cup \{s_3(j) : j = 0, 1\}$. Consider the recursive specification of the process P below:

$$\begin{aligned} P = &\sum_{i=0,1} r_1(i) \cdot \sum_{j=0,1} r_2(j) \cdot s_3(i + j \bmod 2) \cdot P \\ &+ \sum_{l=0,1} r_2(l) \cdot \sum_{k=0,1} r_1(k) \cdot s_3(k + l \bmod 2) \cdot P. \end{aligned}$$

The problem with this specification is that we actually want to denote this process with a merge. For it has parallel input. But if we do so, we will obtain difficulties with the scope of the sum signs. We could write the specification above as follows:

$$P = \Big(\sum_{i=0,1} r_1(i) \parallel \sum_{j=0,1} r_2(j) \Big) \cdot s_3(i + j \bmod 2) \cdot P.$$

Since the variable i under the first sum sign is bounded, we can alter it without changing the meaning of the sum. But the i that occurs in the send action will remain the same. So the meaning of the entire formula will change. We will use an auxiliary unary operator to solve this problem. Let $N = \{n_{k,l} : k, l = 0, 1\}$ be a set of function names. Let $E(N)$ be the following linear functional specification.

$$\begin{aligned} E(N) = &\big\{ n_{k,l}\big(r'_i(j)\big) = r_i(j) : j, k, l = 0, 1\ i = 1, 2 \big\} \\ &\cup \big\{ n_{k,l}(t) = s_3(k + l \bmod 2) : k, l = 0, 1 \big\} \\ &\cup \{n(a) = a : a \in B\} \\ &\cup \big\{ n_{k,l}\big(r'_1(j) \cdot x\big) = r_1(j) \cdot n_{j,l}(x) : j, k, l = 0, 1 \big\} \\ &\cup \big\{ n_{k,l}\big(r'_2(j) \cdot x\big) = r_2(j) \cdot n_{k,j}(x) : j, k, l = 0, 1 \big\} \\ &\cup \big\{ n_{k,l}(t \cdot x) = s_3(k + l \bmod 2) \cdot n_{0,0}(x) : k, l = 0, 1 \big\} \\ &\cup \{n_{k,l}(a \cdot x) = a \cdot n_{k,l}(x) : a \in B\ k, l = 0, 1\}. \end{aligned}$$

With the aid of *ODP*, we know that there is a valuation $\varphi : N \longrightarrow F$ which solves this system. Let us say $\varphi(n_{k,l}) = \nu_{k,l} \in F$ with $k, l \in \{0, 1\}$. Subsequently, we will give a specification of a process X.

$$X = \Big(\sum_{i=0,1} r'_1(i) \parallel \sum_{j=0,1} r'_2(j) \Big) \cdot t \cdot X.$$

With the notations that we have introduced above we can now state the following theorem.

Theorem (6.5)

$$\nu_{0,0}(X) = P.$$

Proof. Consider the following calculation.

$$\begin{aligned} \nu_{0,0}(X) &= \nu_{0,0}\Big(\sum_{i=0,1} r'_1(i) \cdot \sum_{j=0,1} r'_2(j) \cdot t \cdot X \Big) \\ &\quad + \nu_{0,0}\Big(\sum_{l=0,1} r'_2(l) \cdot \sum_{k=0,1} r'_1(k) \cdot t \cdot X \Big) \\ &= \sum_{i=0,1} r_1(i) \cdot \nu_{i,0}\Big(\sum_{j=0,1} r'_2(j) \cdot t \cdot X \Big) \\ &\quad + \sum_{l=0,1} r_2(l) \cdot \nu_{0,l}\Big(\sum_{k=0,1} r'_1(k) \cdot t \cdot X \Big) \end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=0,1} r_1(i) \cdot \sum_{j=0,1} \cdot \nu_{i,j}(t \cdot X) \\
&\quad + \sum_{l=0,1} r_2(l) \cdot \sum_{k=0,1} r_1(k) \cdot \nu_{k,l}(t \cdot X) \\
&= \sum_{i=0,1} r_1(i) \cdot \sum_{j=0,1} \cdot s_3(i + j \bmod 2) \cdot \nu_{0,0}(X) \\
&\quad + \sum_{l=0,1} r_2(l) \cdot \sum_{k=0,1} r_1(k) \cdot s_3(k + l \bmod 2) \cdot \nu_{0,0}(X).
\end{aligned}
$$

With the aid of *RSP* we see that $\nu_{0,0}(X) = P$. This will end the verification of theorem (6.5).

7. Generalizations

In this section we will mention briefly some generalizations of this theory. In particular, we will point out that the definition of a linear functional specification can be generalized easily such that, with the aid of *ODP* and *OSP*, we can introduce more linear unary operators. The first generalization that can be thought of is an "exit" possibility. That is, we allow functional equations of the following type:

$$f(a \cdot x) = f(a).$$

Then we are able to specify (operators that behave like) the projection operators, with the aid of a linear functional specification. The second generalization that we can think of is that we have more "liberal" boundary conditions: we would sometimes send an atomic action to a closed term. To exemplify this, we will give hereinafter a definition of a linear functional specification that accommodates both generalizations.

Definition (7.1)

Let N be a set of function names. A linear functional specification $E(N)$ for N is a set of the following form:

$$E(N) = \{r_{n,a} : n \in N, a \in A\} \cup \{e_{n,a} : n \in N, a \in A\}. \tag{1}$$

Both $r_{n,a}$ and $e_{n,a}$ are equations. Now fix an $a \in A$ and an $n \in N$, then we will define the two equations $r_{n,a}$ and $e_{n,a}$ for the pair (n, a). The first equation $r_{n,a}$ is called a boundary condition and has the following form: there are closed terms $\{t_k : k \in K\}$ and $\{s_l : l \in L\}$ without linear unary operators, for certain finite disjoint sets K and L, such that

$$r_{n,a} \equiv \chi(n, a) = \sum_{k \in K} t_k + \sum_{l \in L} s_l. \tag{2}$$

The equation $e_{n,a}$ is called a linear functional equation and it is of the following form:

$$e_{n,a} \equiv \chi(n, a \cdot x) = \sum_{k \in K} t_k \cdot \chi(n_k, x) + \sum_{l \in L} s_l, \tag{3}$$

in which $\{n_k : k \in K\} \subseteq N$. If $K = \emptyset$, we omit the first sum sign in both equations (2) and (3) and if $L = \emptyset$, we leave out the second sum sign in (2) and (3).

It will be clear that this definition is a generalization of definition (2.2), for let $L = \emptyset$ (this disables the "exit" possibility), let $|K| = 1$ and let the closed term $t_k \in A \cup \{\delta, \tau\}$.

Since, for the generalized state operator*, we have

$$\Lambda_s^m(a) = \sum_{b \in a(m,s)} b,$$

we see that we can specify this operator with a linear functional specification of definition (7.1). We will not need the "exit" possibility here. With this definition, we can also prove theorems on projection operators for open terms. One can think of the following theorem:

Theorem (7.2)

Let $n \geq 1$. Let $H \subseteq A$. Then $\pi_n \circ \partial_H = \partial_H \circ \pi_n$.

Proof. We will make use of the exit possibility here. Let $M = \{m_1, \ldots, m_n\}$ be a set of function names. Consider the following linear functional specification.

$$\begin{aligned} E(M) = &\left\{ m_i(a) = a : a \in A \setminus H, 1 \leq i \leq n \right\} \\ &\cup \left\{ m_i(a) = \delta : a \in H, 1 \leq i \leq n \right\} \\ &\cup \left\{ m_1(a \cdot x) = m_1(a) : a \in A \right\} \\ &\cup \left\{ m_{i+1}(a \cdot x) = m_{i+1}(a) \cdot m_i(x) : a \in A, 1 \leq i < n \right\}. \end{aligned}$$

It will be clear that both $\pi_i \circ \partial_H$ and $\partial_H \circ \pi_i$ $(1 \leq i \leq n)$ are solutions for this system of equations $E(M)$, so with the aid of *OSP*, we conclude that, in particular, $\partial_H \circ \pi_n = \pi_n \circ \partial_H$. This will end the proof of (7.2).

It is also possible to consider operators $f : P \longrightarrow Q$. Where we have a binary operator "+" on the elements of Q. If we take, for example, $Q = 2^A$ we can define the binary operator + to be the union of sets. We must change the definition of a linear functional specification in a radical way. For we might want to have functional equations that look like

$$f(a \cdot x) = f(a) + f(x).$$

Examples of such operators are: the trace operator and the alphabet of a process, which can be found in [**2**]. An example of an auxiliary operator of this "type" can be found in [**18**]. It is the collection of all the registers that occur in a process x and it is abbreviated: reg(x). We will not discuss this type of generalization any further.

Concluding, we can say that the notion of a linear functional specification can be generalized easily, such that we can handle more and more linear unary operators.

---

* see [**2**]

8. Conclusions

In this paper we have presented a way to introduce auxiliary linear unary operators. Moreover, $ACP_{\tau,u}$ is a first attempt to unify many other axiom systems. In section 6, all the examples were originally specified in different axiom systems. While in this paper, these examples are studied in $ACP_{\tau,u}$.

The first example concerning $KFAR_n$, was specified in ACP_τ with renaming operators. This example showed us that the principles *ODP* and *OSP* were not only used to specify an auxiliary linear unary operator in order to give a verification, but we also employed it indirectly: we intensively referred to some (absorption) theorems that were proved, using the principles *ODP* and *OSP*. Another matter that we discovered in studying this example, was that the notion of a guarded recursive specification had to be adapted; see for details remark (2.19).

The second example was specified in *ACP* with renaming operators. These renaming operators were more "elementary" than the renaming operators that were defined in [**17**]. A remarkable fact is that in this example we find linear unary operators inside a guarded recursive specification. As they are so-called concrete operators, they are of no harm in the guarded recursive specification.

The next example uses *ACP* with the localization operator. It is an operator that consists of two parts: there is an encapsulation part and there is an auxiliary part. The localization operator is in fact the composition of a renaming operator and an encapsulation operator. We treated this operator in a slightly different way: we made an auxiliary operator with three components; besides the two mentioned hereinbefore, we adapted an abstraction part, too. This is done to create an example of an abstracting operator and to make things a little more compact.

The last example was originally specified in ACP_τ and the register operator (see [**18**]). To shorten the proof of theorem (6.5), we simplified the situation somewhat: we "reset" the `xor`-port each time after it sent the output along channel 3.

Theorem (4.29) was originally formulated with the aid of *ACP* with the (single) state operator, as can be seen in the remark subsequent to this theorem. We formulate and prove it, in $ACP_{\tau,u}$.

We find thus that $ACP_{\tau,u}$ with of course *ODP* and *OSP*, is a theory that handles all these different cases. So it can be seen as a theory that unifies the other theories. We are not ready, however, since there are examples of auxiliary linear unary operators that we cannot describe with this theory yet. We can mention the generalized state operator here. The main reason for this "lack", is the definition of a linear functional specification. We kept this definition as simple as possible for heuristic reasons.

A matter that we did not discuss here is the research on non-linear unary operators, such as the priority operator; see [**4**]. It is the the author's opinion that it is worthwhile doing some further investigations on both linear and non-linear unary operators.

9. References

[**1**] G. J. Akkerman, J. C. M. Baeten, *Term rewriting analysis in process algebra*, Report P9006, Programming Research Group, University of Amsterdam, 1990.

[**2**] J. C. M. Baeten, J. A. Bergstra, *Global renaming operators in concrete process algebra*, Information and Control **78**, 205–245, 1988.

[**3**] J. C. M. Baeten, J. A. Bergstra, J. W. Klop, *Conditional axioms and α/β-calculus in process algebra*, in: Proc. IFIP Conf. on Formal Description of Programming Concepts III, Ebberup 1986 (M. Wirsing, ed.), North-Holland, Amsterdam.

[**4**] J. C. M. Baeten, J. A. Bergstra, J. W. Klop, *Syntax and defining equations for an interrupt mechanism in process algebra*, Fundamenta Informaticae IX (**2**), pp. 127–168, 1986.

[**5**] J. C. M. Baeten, W. P. Weijland, *Process algebra*, Cambridge university press, 1990.

[**6**] J. A. Bergstra, J. W. Klop, *Algebra of communicating processes with abstraction*, TCS **37**, 77–121, 1985.

[**7**] J. A. Bergstra, J. W. Klop, *Fixed point semantics in process algebras*, MC report IW **206**, 1982. Revised version: J. A. Bergstra, J. W. Klop, *A convergence theorem in process algebra*, CWI report CS-R8733, 1987.

[**8**] J. A. Bergstra, J. W. Klop, *Process Algebra: specification and verification in bisimulation semantics*, Math. & Comp. Sci. II (M. Hazewinkel, J. K. Lenstra, L. G. L. T. Meertens, eds.), CWI Monograph **4**, North-Holland, pp. 61–94, Amsterdam, 1986.

[**9**] J. A. Bergstra, J. W. Klop, *The algebra of recursively defined processes and the algebra of regular processes*, Proceedings 11th ICALP, Antwerpen 1984 (ed. J. Paredaens), Springer LNCS **172**, pp. 82–95, 1984.

[**10**] J. A. Bergstra, J. W. Klop, *Verification of an alternating bit protocol by means of process algebra*, In: Math. Methods of Spec. and Synthesis of Software Systems 1985 (eds. W. Bibel, K. P. Jantke), Math. Research **31**, Akademie-Verlag Berlin, pp. 9–23, 1985.

[**11**] C. C. Chang, H. J. Keisler, *Model Theory*, Second Edition, North-Holland, Amsterdam, 1973.

[**12**] N. Dershowitz, *Termination of rewriting*, Journal of Symbolic Computation, **3**(1), pp. 69–116, 1987.

[**13**] S. Kamin, J.-J. Lévy, *Two generalizations of the recursive path ordering*, Unpublished note, Department of Computer Science, University of Illinois, Urbana, IL, 1980.

[**14**] J. W. Klop, *Term Rewriting Systems*, In: Handbook of Logic in Computer Science (eds. S. Abramsky, D. Gabbay and T. Maibaum), Oxford University Press, To appear.

[**15**] E. Kranakis, *Fixed point equations with parameters in the projective limit model*, Inf. & Comp. **75**, pp. 264–288.

[**16**] K. Meinke, J. V. Tucker, *Universal Algebra*, Oxford University Press, To appear.

[**17**] F. W. Vaandrager, *Verification of two communication protocols by means of process algebra*, report CS-R8608, Centre for Mathematics and Computer Science, Amsterdam, 1986.

[**18**] C. Verhoef, *On the register operator*, Report P9003, Programming Research Group, University of Amsterdam, 1990.

10. Index

In this section we will provide a short index. We will refer to the actual *definition* of a concept and not to the “informal” introducing text that often precedes it. Definitions given on the fly are included, too.