On Induction Principles

C. Verhoef

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

Abstract. We will present a theory that is a generalization of a theory that has been developed to obtain a uniform approach for the introduction of linear unary operators in process algebras. Instances of such operators are frequently found in specifications and verifications of, for example, communication protocols. In many cases a new verification demands a new set of operators. So for each new verification a new axiom system must be given. Moreover, a number of lemmas concerning the brand new operators have to be proved, or, if that fails, additional axioms on these operators have to be added. Although these operators are all different, they share certain common properties, which calls for a unifying framework in which it will be possible to introduce these operators and to reason about them in order to save time and effort. In this paper we will introduce this theory and we will prove some general theorems on linear unary operators. Moreover, we will apply it by proving that a fair FIFO queue satisfies a criterion for protocol correctness. This is not an individual case: we will show that a wide range of communication protocols in which unbounded queues appear can be verified using the same methods.

Key Words & Phrases: *Process Algebras, Algebra of Communicating Processes (with abstraction), Linear Unary Operators, Concurrency, Operator Definition Principle, Operator Specification Principle, Generalized Induction Principle, Unification of Process Algebras, FIFO queues, protocol verification, Time out, Faulty queue, Stuttering queue, Faulty and stuttering queue.*

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.*

1. Introduction

We will consider the algebra of communicating processes with abstraction and linear unary operators. The abbreviation for this axiom system will be $ACP_{\tau,u}$. The τ stands for Milner's silent action [**16**] and the u for unary (in linear unary operator). The theory $ACP_{\tau,u}$ is an extension of the algebra of communicating processes with abstraction ACP_τ that has been studied in [**4**]. In this paper we will generalize the work that has been carried out in [**18**]. In that paper the idea of a two sorted algebra is proposed and it is shown that this is a proper way to obtain a uniform approach for the introduction of linear unary operators and moreover, a number of examples illustrate that $ACP_{\tau,u}$ is a unifying framework for many other theories that consist of *ACP* or ACP_τ with some additional linear unary operators.

In this paper we study a small generalization of the theory that has been introduced in [**18**]: we are now able to specify the projection operators, as well. The projection operators can be found for the first time in process algebra in [**6**]. Due to the systematic research on linear unary operators more general projection operators are proposed instead of the "classical" or *full* projection operators. An induction principle called the approximation induction principle (*AIP*) accompanies the full projection operators. In this paper we propose its equivalent for the generalized projection operators. In fact, we propose a number of induction principles: one for each subset of the finite set of atomic actions. If we take this subset to be the set of atomic actions itself we will regain the old induction principle *AIP*. We will show that these principles can be very useful in verifications of correctness theorems on communication protocols.

We will briefly mention the overall structure of this paper. In the next section we will repeat the main points of the axiomatic framework $ACP_{\tau,u}$ as it is already described in [**18**] albeit that certain matters are generalizations of the corresponding ones in [**18**]. In the subsequent section 3 we will prove some general theorems on linear unary operators. The items in this section are both a continuation and a generalization of theorems that have been studied in [**18**]. In this section we will emphasize on theorems in combination with the projection operators; the reader interested in general theorems without projection operators is referred to [**18**]. An important theorem in this section is the result that the abstraction operator and the projection operators commute under certain circumstances. The proof of this theorem is trivial but its consequences not. In section 4 we will see this when we will explain how this theory can be used in protocol verifications. We will show that a fair FIFO queue with an unbounded capacity satisfies an algebraic criterion for protocol correctness; this is an old problem in process algebra and it has been first noticed in [**5**]. With the methods that we use in this proof we will verify three alternating bit protocols with a time out that can take place at any moment. In the next section we will state some conclusions and recommendations for the further development of this theory and we mention a number of protocols that can be verified in the same way as the ones in section 4. The last section consist of an enumeration of references. In most cases we refer to the first source in which an item can be found. However, this is not always the best possible reference. General references to the subject of process algebra and its applications are [**3**] for the process algebra and [**1**] for several applications.

2. Definitions

In this section we will give the signature and the axioms of $ACP_{\tau,u}$. We will formulate six principles (with accompanying definitions): the Operator Definition Principle *ODP*, the Operator Specification Principle *OSP*, the Recursive Definition Principle *RDP*, the Recursive Specification Principle *RSP*, the Generalized Induction Principle *GIP* and Koomen's Fair Abstraction Rule $KFAR_n$. The first two *ODP* and *OSP* can be found in [**18**]. The principles *RDP* and *RSP* can already be found in [**5**]. The principle *GIP* is new here and is the subject of research in this paper. It is a generalization of the Approximation Induction Principle AIP^- that can be found in [**10**]; see also [**7**]. Finally, $KFAR_n$ can be found already in [**8**]. *RDP*, *RSP* and *KFAR* were introduced in papers about protocol verification. *AIP* is not used in protocol verifications. The reason for that is that the operators occurring in *AIP* do not communicate with an important linear unary operator: the abstraction operator. Using *GIP* will by-pass this problem. We will show this in the next section (see theorem (3.12)).

First we will provide a picture of the signature of $ACP_{\tau,u}$.

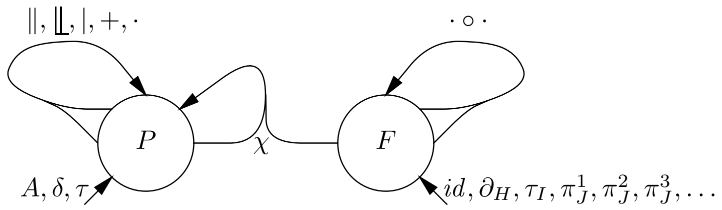

Figure 1. Graphical representation of the signature.

Now we will explain this figure. As we can see it is a two-sorted algebra. The first sort that we consider is the sort P of processes. We have a set of constants or atomic actions A in P. There are two so-called special constants in P: deadlock, which is denoted by δ and the silent step, which is abbreviated by τ. We have five binary operators with both arguments in P (they are all infix operators):

$$\begin{array}{ll}
\text{merge:} & \| : P \times P \longrightarrow P, \\
\text{left-merge:} & \mathbin{|\!\!\lfloor} : P \times P \longrightarrow P, \\
\text{communication-merge:} & | : P \times P \longrightarrow P, \\
\text{sequential composition:} & \cdot : P \times P \longrightarrow P, \\
\text{alternative composition:} & + : P \times P \longrightarrow P.
\end{array}$$

Now we will consider the second one: the sort F of functions or linear unary operators. At first we will discuss the constants of this sort. We have a constant id, which stands for the identity. For all $H \subseteq A$ we have a constant $\partial_H \in F$. We will refer to this constant as the encapsulation operator. For all $I \subseteq A$ we have a constant $\tau_I \in F$, which is called the abstraction operator. Furthermore, we have for all $n \geq 1$ and for all $J \subseteq A$ a constant $\pi^n_J \in F$. This constant will be called the (nth) projection operator. We have only one binary operator with both arguments of sort F:

$$\circ : F \times F \longrightarrow F.$$

This operator will be called the composition of functions. Finally, we have a binary operator that "connects" the two sorts P and F. It is called the apply function:

$$\chi : F \times P \longrightarrow P.$$

This will end the discussion on the signature. Afterwards, we will give the axiom system of $ACP_{\tau,u}$ as it is stated in [**18**] albeit that we will have other axioms concerning the projection operators. We will use the following notational conventions for the symbols that appear in table 1 (page 4): a, b and c are atomic actions, or δ; x, y, z are processes; γ is a special constant, that is, deadlock or silent step; $n \geq 1$; H, I and J are subsets of A; and finally, f, g and h are linear unary operators.

We will formulate an extensionality axiom, which states that linear unary operators that coincide on all processes are indeed the same. We will not use this axiom since we will introduce the notion of a linear functional specification and some principles that allow us to reason about linear unary operators in a comfortable way, but there is no reason to exclude extensionality, so here it is anyway.

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.

Hereinafter, we will define the notion of a linear functional specification. This concept is already defined in [**18**]. We will use a slightly generalized form in which it will be possible to specify the "generalized" projection operators $\pi^n_J \in F$.

$x+y=y+x$	A1	$\chi(\tau_I,a)=a,\quad \text{if } a\notin I$	TI1
$x+(y+z)=(x+y)+z$	A2	$\chi(\tau_I,a)=\tau,\quad \text{if } a\in I$	TI2
$x+x=x$	A3	$\chi(\tau_I,x\cdot y)=\chi(\tau_I,x)\cdot\chi(\tau_I,y)$	TI3
$(x+y)\cdot z=x\cdot z+y\cdot z$	A4		
$(x\cdot y)\cdot z=x\cdot(y\cdot z)$	A5	$\tau \mid x=\delta$	TC1
$x+\delta=x$	A6	$x \mid \tau=\delta$	TC2
$\delta\cdot x=\delta$	A7	$(\tau\cdot x)\mid y=x\mid y$	TC3
		$x\mid(\tau\cdot y)=x\mid y$	TC4
$a\mid b=b\mid a$	C1		
$(a\mid b)\mid c=a\mid(b\mid c)$	C2	$x\cdot\tau=x$	T1
$\delta\mid a=\delta$	C3	$\tau\cdot x+x=\tau\cdot x$	T2
		$a\cdot(\tau\cdot x+y)=a\cdot(\tau\cdot x+y)+a\cdot x$	T3
$x\parallel y=x\mathbin{\|\!\_}y+y\mathbin{\|\!\_}x+x\mid y$	CM1		
$a\mathbin{\|\!\_}x=a\cdot x$	CM2	$\tau\mathbin{\|\!\_}x=\tau\cdot x$	TM1
$(a\cdot x)\mathbin{\|\!\_}y=a\cdot(x\parallel y)$	CM3	$(\tau\cdot x)\mathbin{\|\!\_}y=\tau\cdot(x\parallel y)$	TM2
$(x+y)\mathbin{\|\!\_}z=x\mathbin{\|\!\_}z+y\mathbin{\|\!\_}z$	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		
$(x+y)\mid z=x\mid z+y\mid z$	CM8	$\chi(f,\gamma)=\gamma$	X1
$x\mid(y+z)=x\mid y+x\mid z$	CM9	$\chi(f,\gamma\cdot x)=\gamma\cdot\chi(f,x)$	X2
		$\chi(f,x+y)=\chi(f,x)+\chi(f,y)$	X3
$\chi(\partial_H,a)=a,\quad \text{if } a\notin H$	D1		
$\chi(\partial_H,a)=\delta,\quad \text{if } a\in H$	D2	$\chi(\pi_J^n,a)=a$	GPR1
$\chi(\partial_H,x\cdot y)=\chi(\partial_H,x)\cdot\chi(\partial_H,y)$	D3	$\chi(\pi_J^1,a\cdot x)=a,\quad \text{if } a\in J$	GPR2
		$\chi(\pi_J^{n+1},a\cdot x)=a\cdot\chi(\pi_J^n,x),\quad \text{if } a\in J$	GPR3
$\chi(id,x)=x$	ID	$\chi(\pi_J^n,a\cdot x)=a\cdot\chi(\pi_J^n,x),\quad \text{if } a\notin J$	GPR4

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

Definition (2.2) *Linear Functional Specifications*

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

$$E(N)=\big\{\, r_{n,a} | n\in N, a\in A\,\big\}\cup\big\{\, e_{n,a} | n\in N, a\in A\,\big\},$$

in which $r_{n,a}$ is called a boundary condition and $e_{n,a}$ is called a (linear) functional equation. Fix $a\in A$ and $n\in N$. Then we will define what the form is of these two types of equations. First we define the boundary condition. There is a $b\in A\cup\{\delta,\tau\}$ such that

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

Now we will define what the functional equation is. It can have two forms. We will use the same b

as in the boundary condition that we defined above.

$$e_{n,a} \equiv \begin{cases} \chi(n, a \cdot x) = b, & \text{or} \\ \chi(n, a \cdot x) = b \cdot \chi(m, x), & \text{for one } m \in N. \end{cases}$$

In [**18**] we have only the second form for the functional equation.

Subsequently, we will formulate two principles, which say that a linear functional specification has a solution and that a linear functional specification has at most one solution. Therefore, we first will need the definition of a valuation.

Definition (2.3)

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

Principle (2.4) *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 (2.5) *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*.

Hereinafter, we will introduce two other principles that can be found in [**6**]. But first we will need some preliminary definitions. The following two definitions are taken from [**18**].

Definition (2.6)

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

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

(*ii*) $\big(g \in F,\, a \in A,\, b \in A \cup \{\tau\},\, \forall x \in P : f(a \cdot x) = b \cdot g(x)\big) \Longrightarrow g \in D(f)$

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

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

Remarks (2.7)

We left out δ in the equation $f(a \cdot x) = b \cdot g(x)$. We did that since we otherwise would have $D(\partial_H) = F$. For, we find for all $f \in F : \partial_H(h \cdot x) = \delta = \delta \cdot f(x)$. This would imply that $f \in D(\partial_H)$, thus: $F = D(\partial_H)$. Without the δ we find $D(\partial_H) = \{\partial_H\}$. And this is exactly what it should be: the set of all the operators that ∂_H can become after "passing" the right atomic actions. It is also very easy to see that $D(\pi^1_J) = \{\pi^1_J\}$. Assume that there is a linear unary operator $f \in F$ such that $f \in D(\pi^1_J)$ and that $f \neq \pi^1_J$. Then there must be an atomic action $j \in J$ such that $\pi^1_J(j \cdot x) = j \cdot f(x)$. Then we find that $f(x) = \tau$ for all $x \in P$. But then we see with X1: $\tau = f(\delta) = \delta$. So we find that $D(\pi^1_J) = \{\pi^1_J\}$.

Definition (2.8)

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 $I \neq \emptyset$).

Definition (2.9)

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 (2.10)

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 (2.11)

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 (2.12) *The Recursive Definition Principle*

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

Principle (2.13) *The Recursive Specification Principle*

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

Principle (2.14) *The Generalized Approximation Induction Principle*

Let $J \subseteq A$ and let $x, y \in P$. If we have for all $n \geq 1 : \pi_J^n(x) = \pi_J^n(y)$ and we have that x or y can be specified with the aid of a guarded recursive specification, we have $x = y$. We will use the abbreviation *GIP* for this principle; this acronym stands for: "generalized induction principle".

Remarks (2.15)

We will explain why this principle is called "generalized". In [**7**] we can find this principle for $J = A$. The projection operators are simply denoted by π_n in there. It has the subsequent form: if we have for all $n \geq 1 : \pi_n(x) = \pi_n(y)$ then we have $x = y$. This principle is called "the approximation induction principle" hence the adjective generalized. We will also use the notation $\pi_n = \pi_A^n$ in accordance with the notational conventions of process algebra.

Now we will introduce another principle. It can be found in [**8**].

Principle (2.16) *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. But if we have that $n = 1$ we will also use the abbreviation *KFAR*.

Remark (2.17)

At present, we are able to explain why we demanded that x or y can be specified with the aid of a guarded recursive specification. It was pointed out in [**10**] that

$$\mathit{ACP} + \mathit{RDP} + \mathit{AIP} + \mathit{CA} + \mathit{KFAR} \vdash \tau + \tau \cdot \delta. \tag{1}$$

In other words the combination of these axioms is inconsistent. To solve this problem a more restrictive form of the approximation induction principle (AIP^-) is proposed in there. We will use this idea in a simplified way: we require that x or y can be specified with the aid of a guarded recursive specification. If we look closely at equation (1), we see that we did not met *CA* yet. This abbreviation stands for conditional axioms. The particular conditional axiom that has been used to prove equation (1) is the one that states:

$$\forall\, J_1, J_2 \subseteq A \Longrightarrow \tau_{J_1} \circ \tau_{J_2} = \tau_{J_1 \cup J_2}.$$

With the aid of *OSP* it is very easy to prove this axiom. This is already done in [**18**].

3. Theorems

In this section we will prove some general theorems on linear unary operators. We will emphasize on theorems in which the projection operators occur. Theorems concerning only linear unary operators without an exit possibility can be found in [**18**]. The first result is that the projection operators commute. The second result is that in certain circumstances the composition of two projections can be written as one projection. This is generally true for full projections (which is the next result). Then a number of results follow in which necessary conditions are formulated such that a linear unary operator without an exit possibility commutes with a projection operator. This will be generalized to linear unary operators with an exit. The last theorem states that a linear unary operator with an exit possibility can be decomposed in a linear unary operator without an exit and a generalized projection operator. This is an important result. In [**18**] we discussed the linear unary operators without an exit in depth and in this section we will discuss the projections. This theorem states that we can investigate linear unary operators as they are defined in here by researching two simpler classes: the ones in [**18**] and merely the projections.

Theorem (3.1)

For all $n, k \geq 1$ and $J_1, J_2 \subseteq A$ we have $\pi^n_{J_1} \circ \pi^k_{J_2} = \pi^k_{J_2} \circ \pi^n_{J_1}$.

Proof. Let $M = \{m_{i,j} : 1 \leq i \leq n, 1 \leq j \leq k\}$ be a set of function names. Consider the following linear functional specification:

$$\begin{aligned} E(M) = &\left\{ m_{i,j}(a) = a : a \in A, 1 \leq i \leq n, 1 \leq j \leq k \right\} \\ &\cup \left\{ m_{i,j}(a \cdot x) = m_{i,j}(a) \cdot m_{i,j}(x) : a \in A \setminus (J_1 \cup J_2), 1 \leq i \leq n, 1 \leq j \leq k \right\} \\ &\cup \left\{ m_{1,j}(a \cdot x) = m_{1,j}(a) : a \in J_1, 1 \leq j \leq k \right\} \\ &\cup \left\{ m_{i,1}(a \cdot x) = m_{i,1}(a) : a \in J_2, 1 \leq i \leq n \right\} \\ &\cup \left\{ m_{i+1,j}(a \cdot x) = m_{i+1,j}(a) \cdot m_{i,j}(x) : a \in J_1 \setminus J_2, 1 \leq i < n, 1 \leq j \leq k \right\} \\ &\cup \left\{ m_{i,j+1}(a \cdot x) = m_{i,j+1}(a) \cdot m_{i,j}(x) : a \in J_2 \setminus J_1, 1 \leq i \leq n, 1 \leq j < k \right\} \\ &\cup \left\{ m_{i+1,j+1}(a \cdot x) = m_{i+1,j+1}(a) \cdot m_{i,j}(x) : a \in J_1 \cap J_2 1 \leq i < n, 1 \leq j < k \right\}. \end{aligned}$$

We will try the solution $m_{i,j} = \pi^i_{J_1} \circ \pi^j_{J_2}$. First, we will verify that the boundary conditions hold. Because of GPR1 we immediately see that $\pi^i_{J_1} \circ \pi^j_{J_2}(a) = a$ for all $a \in A$ and $1 \leq i \leq n, 1 \leq j \leq k$. Subsequently, we will check that the functional equations hold. Let thereunto $a \in A \setminus (J_1 \cup J_2)$ and let $1 \leq i \leq n, 1 \leq j \leq k$ be chosen. Then we see, using GPR4 twice, that

$$\pi^i_{J_1} \circ \pi^j_{J_2}(a \cdot x) = \pi^i_{J_1} \circ \pi^j_{J_2}(a) \cdot \pi^i_{J_1} \circ \pi^j_{J_2}(x).$$

Now we treat the case $a \in J_1, 1 \leq j \leq k$. In fact, we have two subcases here: $a \in J_2$ and $a \notin J_2$. But eventually the remaining part will be "cut off" by $\pi^1_{J_1}$. So we obtain that

$$\pi^1_{J_1} \circ \pi^j_{J_2}(a \cdot x) = \pi^i_{J_1} \circ \pi^j_{J_2}(a).$$

The case $a \in J_2, 1 \leq i \leq n$ is treated similar to this case but more simple. Now we take $a \in J_1 \setminus J_2$ and $1 \leq i < n, 1 \leq j \leq k$. We find with the aid of the axioms for the projection operators at once that

$$\pi^{i+1}_{J_1} \circ \pi^j_{J_2}(a \cdot x) = \pi^{i+1}_{J_1} \circ \pi^j_{J_2}(a) \cdot \pi^i_{J_1} \circ \pi^j_{J_2}(x).$$

The case $a \in J_2 \setminus J_1, 1 \leq i \leq n, 1 \leq j < k$ is verified the same as the former case. Finally, we will treat the case $a \in J_1 \cap J_2, 1 \leq i < n, 1 \leq j < k$. We find immediately that

$$\pi^{i+1}_{J_1} \circ \pi^{j+1}_{J_2}(a \cdot x) = \pi^{i+1}_{J_1} \circ \pi^{j+1}_{J_2}(a) \cdot \pi^i_{J_1} \circ \pi^j_{J_2}(x).$$

Now we can conclude that the solution that we proposed is indeed a solution for $E(M)$. On the other hand, if we try the solution $m_{i,j} = \pi^j_{J_2} \circ \pi^i_{J_1}$, we obtain, just as above, that this will solve the system $E(M)$, too. Thus, we may conclude in accord with *OSP* that these solutions are the same. In particular, we find that $\pi^n_{J_1} \circ \pi^k_{J_2} = \pi^k_{J_2} \circ \pi^n_{J_1}$. Herewith we end the verification of (3.1).

Theorem (3.2)

For all $k \geq n \geq 1$ and for all subsets $J_1, J_2 \subseteq A$ with $J_2 \subseteq J_1$ we have $\pi^n_{J_1} \circ \pi^k_{J_2} = \pi^n_{J_1}$.

Proof. We will prove (3.2) with induction on n. Hence, let $n = 1, k \geq 1$ and $J_2 \subseteq J_1 \subseteq A$ be chosen. Let m be a function name. Consider the linear functional specification $E(m)$, defining $\pi^1_{J_1}$, hereinafter.

$$\begin{aligned} E(m) = &\left\{ m(a) = a : a \in A \right\} \\ &\cup \left\{ m(a \cdot x) = m(a) : a \in J_1 \right\} \\ &\cup \left\{ m(a \cdot x) = m(a) \cdot m(x) : a \in A \setminus J_1 \right\}. \end{aligned}$$

We immediately see that $\pi^1_{J_1}$ is a solution for $E(m)$. We will show that $\pi^1_{J_1} \circ \pi^k_{J_2}$ is also a solution. Thereunto we will verify that the boundary conditions hold. Let $a \in A$ be fixed. Then we see with GPR1 that $\pi^1_{J_1} \circ \pi^k_{J_2}(a) = a$. Now we will consider the functional equations. First let $a \in J_1$. We will have two cases: $a \notin J_2$ and $a \in J_2$. The latter case will have two subcases: $k = 1$ and $k > 1$. Consider the following simple calculation:

$$\begin{aligned}\pi^1_{J_1} \circ \pi^k_{J_2}(a \cdot x) &= \begin{cases} a \in J_2: & \begin{cases} \pi^1_{J_1} \circ \pi^k_{J_2}(a), & \text{if } k = 1; \\ \pi^1_{J_1}\big(\pi^k_{J_2}(a) \cdot \pi^{k-1}_{J_2}(x)\big), & \text{if } k > 1. \end{cases} \\ & \\ a \notin J_2: & \pi^1_{J_1}\big(\pi^k_{J_2}(a) \cdot \pi^k_{J_2}(x)\big) \end{cases} \\ &= \pi^1_{J_1} \circ \pi^k_{J_2}(a).\end{aligned}$$

Now we will treat the case $a \in A \setminus J_1 \subseteq A \setminus J_2$. It is evident that

$$\pi^1_{J_1} \circ \pi^k_{J_2}(a \cdot x) = \pi^1_{J_1} \circ \pi^k_{J_2}(a) \cdot \pi^1_{J_1} \circ \pi^k_{J_2}(x).$$

Thus, we find, using *OSP*:

$$\forall k \geq 1,\, \forall J_2 \subseteq J_1 : \pi^1_{J_1} \circ \pi^k_{J_2} = \pi^1_{J_1}. \tag{1}$$

Now suppose that (3.2) is proved up to n inclusive. Then we will verify (3.2) for $n+1$. Choose $k \geq n+1$ and let $J_2 \subseteq J_1 \subseteq A$ be fixed. Let $M = \{m_1, \ldots, m_{n+1}\}$ be a set of function names and consider the subsequent linear functional specification $E(M)$:

$$\begin{aligned} E(M) = &\left\{ m_i(a) = a : a \in A,\, 1 \leq i \leq n+1 \right\} \\ &\cup \left\{ m_i(a \cdot x) = m_i(a) \cdot m_i(x) : a \in A \setminus J_1,\, 1 \leq i \leq n+1 \right\} \\ &\cup \left\{ m_{i+1}(a \cdot x) = m_{i+1}(a) \cdot m_i(x) : a \in J_1,\, 1 \leq i \leq n \right\} \\ &\cup \left\{ m_1(a \cdot x) = m_1(a) : a \in J_1 \right\}. \end{aligned}$$

It will be clear that $\pi^1_{J_1}, \ldots, \pi^{n+1}_{J_1}$ solves the linear functional specification $E(M)$. We will try the solution $m_i = \pi^i_{J_1} \circ \pi^k_{J_2}$ for $1 \leq i \leq n+1$. It is obvious that the boundary conditions are satisfied. So we will only treat the functional equations. Let $a \in A \setminus J_1$ and $1 \leq i \leq n+1$. Since $a \notin J_2$ we find:

$$\pi^i_{J_1} \circ \pi^k_{J_2}(a \cdot x) = \pi^i_{J_1} \circ \pi^k_{J_2}(a) \cdot \pi^i_{J_1} \circ \pi^k_{J_2}(x).$$

Now let $a \in J_1$. First we will handle the case $i = 1$.

$$\begin{aligned} \pi^1_{J_1} \circ \pi^k_{J_2}(a \cdot x) &= \pi^1_{J_1}(a \cdot x) && \text{because of (1)} \\ &= \pi^1_{J_1}(a) \\ &= \pi^1_{J_1} \circ \pi^k_{J_2}(a). \end{aligned}$$

Now let $1 \leq i < n$. Consider the following calculation.

$$\begin{aligned} \pi^{i+1}_{J_1} \circ \pi^k_{J_2}(a \cdot x) &= \pi^{i+1}_{J_1}(a \cdot x) && \text{induction hypothesis} \\ &= \pi^{i+1}_{J_1}(a) \cdot \pi^i_{J_1}(x) \\ &= \pi^{i+1}_{J_1} \circ \pi^k_{J_2}(a) \cdot \pi^i_{J_1}(x) \\ &= \pi^{i+1}_{J_1} \circ \pi^k_{J_2}(a) \cdot \pi^i_{J_1} \circ \pi^k_{J_2}(x). && \text{induction hypothesis} \end{aligned}$$

Now we will treat the case $i = n+1$. We will have two subcases: $a \in J_2$ and $a \notin J_2$. First, we will consider $a \in J_2$ (observe that $k > 1$):

$$\begin{aligned} \pi^{n+1}_{J_1} \circ \pi^k_{J_2}(a \cdot x) &= \pi^{n+1}_{J_1}\big(\pi^k_{J_2}(a) \cdot \pi^{k-1}_{J_2}(x)\big) \\ &= \pi^{n+1}_{J_1} \circ \pi^k_{J_2}(a) \cdot \pi^n_{J_1} \circ \pi^{k-1}_{J_2}(x) \\ &= \pi^{n+1}_{J_1} \circ \pi^k_{J_2}(a) \cdot \pi^n_{J_1}(x) && \text{induction hypothesis} \\ &= \pi^{n+1}_{J_1} \circ \pi^k_{J_2}(a) \cdot \pi^n_{J_1} \circ \pi^k_{J_2}(x). && \text{induction hypothesis} \end{aligned}$$

Now we take $a \notin J_2$.

$$\begin{aligned}\pi^{n+1}_{J_1} \circ \pi^k_{J_2}(a \cdot x) &= \pi^{n+1}_{J_1}\big(\pi^k_{J_2}(a) \cdot \pi^k_{J_2}(x)\big)\\ &= \pi^{n+1}_{J_1} \circ \pi^k_{J_2}(a) \cdot \pi^n_{J_1} \circ \pi^k_{J_2}(x).\end{aligned}$$

We see that in both subcases the right-hand sides are the same. Hence, we see that the functional equations are satisfied, too. So with the aid of *OSP* we find that $\pi^{n+1}_{J_1} \circ \pi^k_{J_2} = \pi^{n+1}_{J_1}$. This will end the induction step and therewith the proof of (3.2).

Corollary (3.3)

For all $n, k \geq 1$ and for all $J \subseteq A$ we have $\pi^n_J \circ \pi^k_J = \pi^{\min(n,k)}_J$.

Proof. Let $J_1 = J_2 = J$. First suppose that $k \geq n$. With the aid of theorem (3.2) we find that

$$\pi^n_J \circ \pi^k_J = \pi^n_J = \pi^{\min(n,k)}_J.$$

Now suppose that $k < n$. Then we find:

$$\begin{aligned}\pi^n_J \circ \pi^k_J &= \pi^k_J \circ \pi^n_J && \text{because of (3.1)}\\ &= \pi^k_J && \text{theorem (3.2)}\\ &= \pi^{\min(n,k)}_J.\end{aligned}$$

This will end the proof of (3.3).

Corollary (3.4)

For all $n, k \geq 1 : \pi_n \circ \pi_k = \pi_{\min(n,k)}$. We used here the abbreviation $\pi_n = \pi^n_A$, see (2.15).

Remark (3.5)

It is a well-known fact in process algebra that for closed terms t the following holds:

$$\pi_n \circ \pi_k(t) = \pi_{\min(n,k)}(t).$$

Confer [**3**]. We see that with the use of linear functional specifications, in combination with *ODP* and *OSP*, we are able to prove this fact for all processes. So a composition of "full" projections is always decomposable in only one full projection. This decomposability result is generalized in theorem (3.2). The following question may arise: "Is every composition of generalized projection operators decomposable in only one (generalized) projection operator?" The answer to this question is negative. For, let $A = \{a, b\}$ and suppose that there exists a subset $J \subseteq A$ and a natural number $n \geq 1$ such that $\pi^1_a \circ \pi^2_b = \pi^n_J$. Assume that $b \notin J$. Then we see that

$$\pi^1_a \circ \pi^2_b(b^3) = b^2 \neq b^3 = \pi^n_J(b^3).$$

So this assumption cannot hold. Hence, $b \in J$. In a similar way we find that $a \in J$, so $J = A$. Assume that $n = 1$. Then we see that

$$\pi^1_J(b^3) = b \neq b^2 = \pi^1_a \circ \pi^2_b(b^3).$$

Thus, we find that $n > 1$. Thus, $\pi^n_J(a^2) = a^2$. But

$$\pi^n_J(a^2) = \pi^1_a \circ \pi^2_b(a^2) = a.$$

So $n \not> 1$. This is in contradiction with the fact that $n > 1$ so the assumption that $\pi^1_a \circ \pi^2_b$ can be written as π^n_J for one $n \geq 1$ and a subset $J \subseteq A$ cannot hold.

We will define hereinafter what a renaming operator is. In [**18**] we define this to be a linear unary operator that can be defined with the aid of a linear functional specification with the extra requirement that the set of derived operators $D(f)$ contains only one element. But with the generalized version of a linear functional specification, we are able to specify π_1. This means that with this definition π_1 would become a renaming operator. To solve this, we will need the subsequent definition in which we will introduce the concept of the set of "exits" of a linear unary operator.

Definition (3.6)

Let $f \in F$ be a linear unary operator. The set of exits $X(f)$ with respect to f is the following subset of A:

$$X(f) = \{\, a \in A \mid \forall x \in P \exists b \in A \cup \{\tau\} : f(a \cdot x) = b \,\}$$

We excluded δ here, since $\delta = \delta \cdot f(x)$ for all $f \in F$. This means that if we have an atomic action a such that $f(a \cdot x) = \delta$, it will be—on the one hand—an exit for f but on the other hand it is not an exit $f(a \cdot x) = \delta \cdot f(x)$.

Examples (3.7)

It is easy to see that $X(\pi^1_J) = J$. For, let $j \in J$ then we have for all $x \in P : \pi^1_J(j \cdot x) = j$ so we see that $j \in X(\pi^1_J)$. Now suppose that $a \notin J$. Then we know that for all $x \in P : \pi^1_J(a \cdot x) = a \cdot \pi^1_J(x)$ and we see that $a \notin X(\pi^1_J)$. Thus, we find that $X(\pi^1_J) = J$. We will show that $X(\partial_H) = \emptyset$. Suppose that there is an atomic action $a \in X(\partial_H)$. If $a \notin H$ we find that $\partial_H(a \cdot x) = a \cdot \partial_H(x)$. This yields that $a \notin X(\partial_H)$. So we must have $a \in H$. But then we find: $\partial_H(a \cdot x) = \delta$. So we see, by definition, that $a \notin X(\partial_H)$. We find thus that $X(\partial_H) = \emptyset$. In a similar way we can obtain that $X(\tau_I) = \emptyset$.

Definition (3.8)

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

Examples (3.9)

We will show that ∂_H is a renaming operator. We have seen in (2.7) that $|D(\partial_H)| = 1$. In (3.7) we have seen that $X(\partial_H) = \emptyset$. It is easy to see that ∂_H can be defined with the aid of a linear functional specification, so we find that ∂_H is a renaming operator. In the same way we can obtain that τ_I is a renaming operator. Now suppose that π^1_J is a renaming operator, then π^1_J must be the identity map. We already know that $|D(\pi^1_J)| = 1$; see (2.7). Since π^1_J is a renaming operator, we have $J = X(\pi^1_J) = \emptyset$. Let n be a function name and consider the following linear functional specification.

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

It is very easy to see that both $\pi^1_\emptyset$ and id* are solutions for $E(n)$. Hence, we have $\pi^1_\emptyset = id$.

Theorem (3.10)

Let $f \in F$ be a renaming operator. Let $J \subseteq A$ be chosen. Suppose that the following two conditions are valid

(i) $\quad f(J) \subseteq J \cup \{\delta\}$

(ii) $\quad f(A \setminus J) \subseteq A \setminus J \cup \{\delta\}$

then we have for all $n \geq 1 : f \circ \pi^n_J = \pi^n_J \circ f$. Observe that f is a concrete renaming operator because of the conditions (i) and (ii).

Proof. Choose an $n \geq 1$ and let $M = \{m_1, \ldots, m_n\}$ be a set of function names. Contemplate the subsequent linear functional specification.

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

* See table 1 page 4.

We will try the solution $m_i = \pi_J^i \circ f$ for $1 \leq i \leq n$. It is very easy to see that the boundary conditions are valid, so we will only verify that the functional equations are satisfied. Let $a \in A \setminus J$ and $1 \leq i \leq n$.

$$\begin{aligned}\pi_J^i \circ f(a \cdot x) &= \pi_J^i\big(f(a) \cdot f(x)\big)\\ &= \pi_J^i \circ f(a) \cdot \pi_J^i \circ f(x). \qquad (ii)\end{aligned}$$

Now let $a \in J$. We have two cases: $i = 1$ and $n \geq i > 1$.

$$\begin{aligned}\pi_J^1 \circ f(a \cdot x) &= \pi_J^1\big(f(a) \cdot f(x)\big)\\ &= \pi_J^1 \circ f(a). \qquad (i)\end{aligned}$$

Now the case $i > 1$:

$$\begin{aligned}\pi_J^{i+1} \circ f(a \cdot x) &= \pi_J^{i+1}\big(f(a) \cdot f(x)\big)\\ &= \pi_J^{i+1} \circ f(a) \cdot \pi_J^i \circ f(x). \qquad (i)\end{aligned}$$

So we see that this solves the system $E(M)$. On the other hand if we try the solution $m_i = f \circ \pi_J^i$ for all $1 \leq i \leq n$, we obtain that this is also a solution for $E(M)$. Using *OSP* we find that, in particular, $f \circ \pi_J^n = \pi_J^n \circ f$. This will end the verification of the theorem.

Theorem (3.10) was originally stated in [**18**]. In there, it was not yet possible to reason with projection operators. We obtained that $f \circ \pi_n(x) = \pi_n \circ f(x)$ for so-called concrete processes x.

Corollary (3.11)

For all $n \geq 1$ and for all subsets $H, J \subseteq A$ we have:

$$\pi_J^n \circ \partial_H = \partial_H \circ \pi_J^n.$$

Proof. We will check that the conditions of theorem (3.10) are satisfied. First, we will take a look at condition (i). Since $\partial_H(a) \subseteq \{a, \delta\}$ for all $a \in A$, we see immediately that (i) holds. The same is valid for condition (ii). Thus, we may apply theorem (3.10) and we obtain the desired equation.

Theorem (3.12)

Let $I, J \subseteq A$ and suppose that $I \cap J = \emptyset$. Then we have for all $n \geq 1 : \pi_J^n \circ \tau_I = \tau_I \circ \pi_J^n$.

Proof. Let $n \geq 1$ be chosen. Let $\{m_1, \ldots, m_n\}$ be a set of function names. Consider the following linear functional specification:

$$\begin{aligned}E(m_1, \ldots, m_n) = &\big\{\, m_i(a) = a : a \in A \setminus I,\, 1 \leq i \leq n \,\big\}\\ &\cup \big\{\, m_i(a) = \tau : a \in I,\, 1 \leq i \leq n \,\big\}\\ &\cup \big\{\, m_i(a \cdot x) = m_i(a) \cdot m_i(x) : a \in A \setminus J,\, 1 \leq i \leq n \,\big\}\\ &\cup \big\{\, m_1(a \cdot x) = m_1(a) : a \in J \,\big\}\\ &\cup \big\{\, m_{i+1}(a \cdot x) = m_{i+1}(a) \cdot m_i(x) : a \in J,\, 1 \leq i < n \,\big\}.\end{aligned}$$

It consists of straightforward calculation to see that if we put for all $1 \leq i \leq n$:

$$m_i = \pi_J^i \circ \tau_I,$$

that this solves the system $E(m_1, \ldots, m_n)$. But on the other hand if we put

$$m_i = \tau_I \circ \pi_J^i,$$

for all $1 \leq i \leq n$, we obtain a solution for $E(m_1, \ldots, m_n)$, too. So with the aid of *OSP* we may conclude that these solutions are the very same. In particular we find that $\pi_J^n \circ \tau_I = \tau_I \circ \pi_J^n$. This will end the verification of (3.12).

Theorem (3.13)

Let f be a renaming operator. Let $J \subseteq A$. Suppose that the following conditions hold.

(*i*) $f(J) \subseteq J \cup \{\delta\}$

(*ii*) $f(A \setminus J) \subseteq A \setminus J \cup \{\delta, \tau\}$

Then we have $\pi_J^n \circ f = f \circ \pi_J^n$ for all $n \geq 1$.

Proof. We could give a direct proof of this theorem, but we will give an indirect proof: first, we will show that we can decompose f into a concrete part and an abstraction operator. Thereinafter, we will apply theorems (3.10) and (3.12). Let $I = \{a \in A : f(a) = \tau\}$ be the *abstracting set* of f. Let n be a function name and consider the following linear functional specification.

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

It will be clear that the solution for this linear functional specification is a concrete renaming operator. We will name it g. It is not very difficult to see that it will be sufficient to show that

$$\forall a \in A : f(a) = g \circ \tau_I(a), \tag{2}$$

in order to be able to conclude that $f = g \circ \tau_I$. Since (2) is valid, we find the desired decomposition. We will verify that g satisfies the conditions (*i*) and (*ii*) of theorem (3.10). Let $a \in J$, then we see that $a \notin I$*. So, we find $g(a) = f(a) \in J \cup \{\delta\}$. Now we verify condition (*ii*). Let $a \in A \setminus J$. If $a \in I$, we find, by definition, $g(a) = a \in A \setminus J \cup \{\delta\}$. So let $a \notin I$. Then we find $g(a) = f(a) \in A \setminus J \cup \{\delta, \tau\}$. But $g(a) \neq \tau$, so $g(a) \in A \setminus J \cup \{\delta\}$. It will be clear that we may use theorem (3.12), so we can make the subsequent calculation.

$$\begin{aligned} f \circ \pi_J^n &= g \circ \tau_I \circ \pi_J^n \\ &= g \circ \pi_J^n \circ \tau_I && \text{because of (3.12)} \\ &= \pi_J^n \circ g \circ \tau_I && \text{because of (3.10)} \\ &= \pi_J^n \circ f. \end{aligned}$$

This will end the verification of (3.13).

Remark (3.14)

The decomposition of a renaming operator f into a concrete renaming operator g and an abstraction operator τ_I is not unique. For, let $A = \{a, b\}$ and let $I = \{b\}$. Let f be the solution for $E(n)$:

$$E(n) = \left\{\, n(a) = b, n(b) = \tau \,\right\} \cup \left\{\, n(c \cdot x) = n(c) \cdot n(x) : c \in A \,\right\}.$$

Let g be the solution for $E(m)$:

$$E(m) = \left\{\, m(a) = b, m(b) = b \,\right\} \cup \left\{\, m(c \cdot x) = m(c) \cdot m(x) : c \in A \,\right\}.$$

Then it is easy to see that $f = g \circ \tau_I$ (of course, g is a concrete renaming operator). Let g' be the solution for $E(m')$:

$$E(m') = \left\{\, m'(a) = b, m'(b) = a \,\right\} \cup \left\{\, m'(c \cdot x) = m'(c) \cdot m'(x) : c \in A \,\right\}.$$

We can find easily that $f = g' \circ \tau_I$. Suppose that $g = g'$ then g' is also a solution for $E(m)$. But $g'(b) \neq b$ and we find that $g' \neq g$. So the decomposition is *not* unique.

* We see that $I \cap J = \emptyset$, so the condition of theorem (3.12) is satisfied

Theorem (3.15)

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 of f_1. Let $X(f_j) = \emptyset$ for all $1 \leq j \leq k$. Suppose that the following conditions hold.

(i) $\bigcup_{j=1}^{k} f_j(J) \subseteq J \cup \{\delta\}$

(ii) $\bigcup_{j=1}^{k} f_j(A \setminus J) \subseteq A \setminus J \cup \{\delta, \tau\}$

Then we have $\pi_J^n \circ f_1 = f_1 \circ \pi_J^n$ for all $n \geq 1$.

Proof. Define a map $\sigma : A \times \{1, 2, \ldots, k\} \longrightarrow \{1, 2, \ldots, k\}$ as follows

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

Let $M = \{m_{i,j} : 1 \leq i \leq n,\, 1 \leq j \leq k\}$ be a set of function names and consider the subsequent linear functional specification $E(M)$.

$$\begin{aligned} E(M) = {} & \big\{\, m_{i,j}(a) = f_j(a) : a \in A,\ 1 \leq i \leq n,\ 1 \leq j \leq k \,\big\} \\ & \cup \big\{\, m_{i,j}(a \cdot x) = m_{i,j}(a) \cdot m_{i,\sigma(a,j)}(x) : a \in A \setminus J,\ 1 \leq i \leq n,\ 1 \leq j \leq k \,\big\} \\ & \cup \big\{\, m_{1,j}(a \cdot x) = m_{1,j}(a) : a \in J,\ 1 \leq j \leq k \,\big\} \\ & \cup \big\{\, m_{i+1,j}(a \cdot x) = m_{i+1,j}(a) \cdot m_{i,\sigma(a,j)}(x) : a \in J,\ 1 \leq i < n,\ 1 \leq j \leq k \,\big\}. \end{aligned}$$

It is very easy to see that if we try the solution $\pi_J^i \circ f_j = m_{i,j}$ that this system will solve the linear functional specification $E(M)$. We will need here both conditions. But if we try on the other hand the solution $f_j \circ \pi_J^i = m_{i,j}$ we will effortlessly see that this will solve the linear functional specification, too. So in accord with *OSP* we may conclude that these solutions must be equal. In particular, we find that $\pi_J^n \circ f_1 = f_1 \circ \pi_J^n$ for all $n \geq 1$. This will end the proof of the theorem.

Below, we will generalize theorem (3.15) to linear unary operators that can have an exit.

Theorem (3.16)

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 its set of derived operators. Suppose that the following conditions hold.

(i) $\bigcup_{j=1}^{k} f_j\big(J \setminus X(f_j)\big) \subseteq J \cup \{\delta\}$

(ii) $\bigcup_{j=1}^{k} f_j\big(A \setminus \big(J \cup X(f_j)\big)\big) \subseteq A \setminus J \cup \{\delta, \tau\}$

Then we have $\pi_J^n \circ f_1 = f_1 \circ \pi_J^n$ for all $n \geq 1$.

Proof. Let $M = \{m_{i,j} : 1 \leq i \leq n,\, 1 \leq j \leq k\}$ be a set of function names. Let

$$\sigma : A \times \{1, 2, \ldots, k\} \longrightarrow \{1, 2, \ldots, k\}$$

be the partial* function defined hereinafter.

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

* If $a \in X(f_i)$ then $\sigma(a, i)$ is not defined.

Consider the next linear functional specification $E(M)$.

$$\begin{aligned}
E(M) = \{\, & m_{i,j}(a) = f_j(a) : 1 \le i \le n,\, 1 \le j \le k \,\} \\
& \cup \bigcup_{j=1}^{k} \{\, m_{i,j}(a \cdot x) = m_{i,j}(a) : a \in X(f_j),\, 1 \le i \le n \,\} \\
& \cup \bigcup_{j=1}^{k} \{\, m_{1,j}(a \cdot x) = m_{1,j}(a) : a \in J \setminus X(f_j) \,\} \\
& \cup \bigcup_{j=1}^{k} \{\, m_{i+1,j}(a \cdot x) = m_{i+1,j}(a) \cdot m_{i,\sigma(a,j)}(x) : a \in J \setminus X(f_j),\, 1 \le i < n \,\} \\
& \cup \bigcup_{j=1}^{k} \{\, m_{i,j}(a \cdot x) = m_{i,j}(a) \cdot m_{i,\sigma(a,j)}(x) : a \in A \setminus \big(J \cup X(f_j)\big),\, 1 \le i \le n \,\}.
\end{aligned}$$

It is rather easy to deduce that, if we put $m_{i,j} = \pi^i_J \circ f_j$, this will solve the linear functional specification $E(M)$ (we will need here the conditions stated in the theorem). But if we put, on the other hand, $m_{i,j} = f_j \circ \pi^i_J$, a simple calculation will learn us that this will solve the system $E(M)$, too. Using *OSP* we see immediately that (3.16) is proved.

Definition (3.17)

A linear unary operator $f \in F$ that can be defined with the aid of a linear functional specification has no exits if for all $g \in D(f)$ their sets of exits $X(g)$ are empty.

Theorem (3.18)

Let $f \in F$ be a linear unary operator that can be defined with the aid of a linear functional specification. Then there is a linear unary operator $g \in F$ without exits such that $f = g \circ \pi^1_{X(f)}$. Moreover, the sets of derived operators of f and g have the same cardinality.

Proof. Let $f = f_1$ and let $D(f) = \{f_1, \ldots, f_k\}$ be the set of derived operators of f. Let $N = \{n_1, \ldots, n_k\}$ be a set of function names. Let $\sigma : A \times \{1, 2, \ldots, k\} \longrightarrow \{1, 2, \ldots, k\}$ be the partial map defined as follows.

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

Consider the linear functional specification $E(N)$ below.

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

According to *ODP* there is a solution for this system. Let us say that $g_1, \ldots, g_k$ is a solution for it. We claim that we can take $g = g_1$. For, let $M = \{m_1, \ldots, m_k\}$ be a set of function names. Let $E(M)$ be the defining linear functional specification of f:

$$\begin{aligned}
E(M) = \{\, & m_i(a) = f_i(a) : a \in A,\, 1 \le i \le k \,\} \\
& \cup \bigcup_{i=1}^{k} \{\, m_i(a \cdot x) = m_i(a) : a \in X(f_i) \,\} \\
& \cup \bigcup_{i=1}^{k} \{\, m_i(a \cdot x) = m_i(a) \cdot m_{\sigma(a,i)}(x) : a \in X(f_i) \,\}
\end{aligned}$$

It will be evident that $f_1, \ldots, f_k$ is a solution for this system. It is also very easy to see that

$$g_1 \circ \pi^1_{X(f_1)}, g_2 \circ \pi^1_{X(f_2)}, \ldots, g_k \circ \pi^1_{X(f_k)}$$

is also a solution for it. So with the aid of *OSP* we can conclude that these solutions are the same. In particular, we find that $f = g_1 \circ \pi^1_{X(f)}$. We see that $|D(f)| = |D(g)|$. This will finish the proof of (3.18).

4. Applications

In this section we will apply the theory that we developed. First, we will state a correctness criterion that can be found among others in [**5**]. Then we will show that processes that satisfy other criterions that are common in protocol verification* also satisfy the criterion stated in this paper (we will prove this in two ways). Furthermore, we will show that a FIFO queue with an unbounded capacity satisfies this correctness criterion. We will base ourselves on a (begin of a) proof that can be found in [**5**]. However, as we can see in the preface of [**1**]: the authors of [**5**] have not been able to give a solution to the problems they have put forward. Then we will formulate a lemma on an apparently important linear unary operator that occurs in the proof of the FIFO queue. Afterwards, we will discuss three alternating bit protocols with a time out that can take place at any moment. They only differ in one way: their communication channels. The acknowledgement channel is perfectly reliable in all three cases. This is done to shorten the proofs but is not a serious constraint. In the first protocol the data channel is also reliable: it is a queue with an unbounded capacity, since the time out can occur at any moment. In the second protocol we make things a bit less unrealistic: the queue with an unbounded capacity can lose its last read datum at any moment. In the last protocol we have a queue that can lose or duplicate any datum that is in the queue at any moment.

The following definition is taken from [**5**].

Definition (4.1)

Let D be a set of datum elements. Let P be a process (see figure 2). Let $H \subseteq A$ be the set of all the internal read and send actions and the input read and send actions of the process P. Let $I \subseteq A$ be the set of all the non-trivial communications that can be made with the atomic actions of H. Let D^ω be the cartesian product of infinitely many countable copies of D. Let the input and output channel of P be labeled with the numbers 1, and 2 respectively. If for all $(d_i)_i \in D^\omega$:

$$\tau_I \circ \partial_H\Big(P \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big) = \tau \cdot \prod_{i=1}^{\infty} s_2(d_i),$$

then we will call P a correct (communication) protocol.

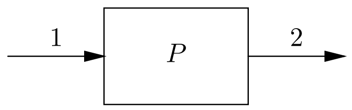

Figure 2. Visualization of the process P.

In protocol verifications we often meet other correctness criteria. One of these criteria is that a protocol is correct if it is bisimilar to a one bit buffer. In other words, if P is our protocol with input channel 1 and output channel 2 then we want that

$$P = \sum_{d \in D} r_1(d) \cdot s_2(d) \cdot P.$$

* See, for instance, [**8**], [**9**] and [**14**].

Another criterion says that the protocol should behave as a two bit buffer. And, more general, we have a criterion that says that our protocol is correct when it behaves as an s-bit buffer. The criterion that we use in this paper says that the protocol must behave like a queue with unbounded capacity (or an "ω-bit buffer").

Hereinafter, we will show that every s-bit buffer satisfies the correctness criterion stated in definition (4.1). Consider thereunto the following visualization of an s-bit buffer.

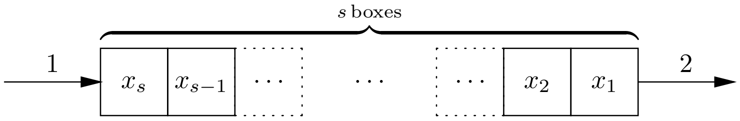

Figure 3. An s-bit buffer with input channel 1 and output channel 2.

Now we will give a guarded recursive specification of an s-bit buffer.

$$B_s = \sum_{x_1 \in D} r_1(x_1) \cdot B_s(x_1)$$

For $1 \le j < s$ we have:

$$B_s(x_1, \ldots, x_j) = \sum_{x_{j+1} \in D} r_1(x_{j+1}) \cdot B_s(x_1, \ldots, x_{j+1}) + s_2(x_1) \cdot B_s(x_2, \ldots, x_j)$$

$$B_s(x_1, \ldots, x_s) = s_2(x_1) \cdot B_s(x_2, \ldots, x_s).$$

Let the following subsets of the set of atomic actions A be given:

$$H = \{\, r_1(x), s_1(x) : x \in D \,\},$$
$$I = \{\, c_1(x) : x \in D \,\}.$$

Below we will prove that B_s is a correct protocol. But first, we will provide a visualization in figure 4 of the left-hand side of equation (1).

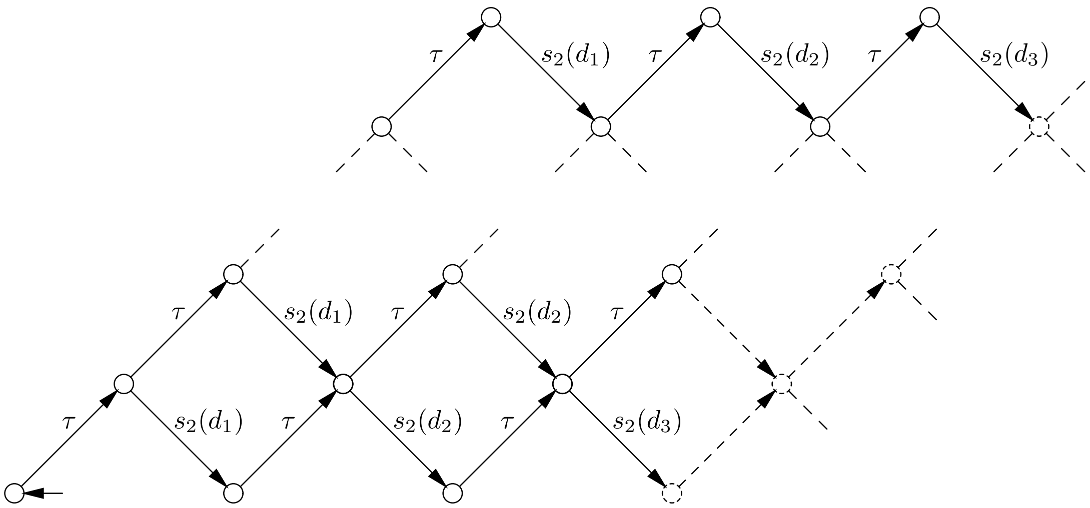

Figure 4. A graph of the left-hand side of equation (1).

Theorem (4.2)

For all $(d_i)_i \in D^\omega$ the following holds:

$$\tau_I \circ \partial_H\Big(B_s \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big) = \tau \cdot \prod_{i=1}^{\infty} s_2(d_i). \tag{1}$$

Proof. Let us choose an element $(d_i)_i \in D^\omega$. We will introduce some notations for convenience sake. For all $k \geq 1$ and $1 \leq j \leq s$ we define

$$\begin{aligned} Z_0^k &= B_s \parallel \prod_{i=k}^{\infty} s_1(d_i) \\ Z_j^k &= B_s(d_k, \ldots, d_{k+j-1}) \parallel \prod_{i=k+j}^{\infty} s_1(d_i). \end{aligned}$$

It is very easy to see that the following holds for all $k \geq 1$ and $1 \leq j < s$:

$$\begin{aligned} &\tau_I \circ \partial_H(Z_0^k) = \tau \cdot \tau_I \circ \partial_H(Z_1^k) \\ &\tau_I \circ \partial_H(Z_j^k) = \tau \cdot \tau_I \circ \partial_H(Z_{j+1}^k) + s_2(d_k) \cdot \tau_I \circ \partial_H(Z_{j-1}^{k+1}) \\ &\tau_I \circ \partial_H(Z_s^k) = s_2(d_k) \cdot \tau_I \circ \partial_H(Z_{s-1}^{k+1}). \end{aligned} \tag{2}$$

We will prove the following claim:

$$(3) \qquad \forall k, n \geq 1, \forall 0 \leq j \leq s : \pi_n \circ \tau_I \circ \partial_H(Z_j^k) = \begin{cases} \tau \cdot \pi_n\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big), & \text{if } 0 \leq j < s; \\ \pi_n\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big), & \text{if } j = s. \end{cases}$$

We will prove (3) with induction on n. So let $n = 1$. Now fix a $k \geq 1$. First, we will take $j = s$. Since

$$\pi_1 \circ \tau_I \circ \partial_H(Z_s^k) = s_2(d_k) = \pi_1\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big),$$

we see that the case $j = s$ is finished. Now suppose that for $1 < j \leq s$ we have proved the following:

$$\pi_1 \circ \tau_I \circ \partial_H(Z_j^k) = \tau \cdot s_2(d_k), \tag{4}$$

then we prove equation (4) for $j - 1$ (observe that we use here axiom T2):

$$\begin{aligned} \pi_1 \circ \tau_I \circ \partial_H(Z_{j-1}^k) &= \tau \cdot \pi_1 \circ \tau_I \circ \partial_H(Z_j^k) + s_2(d_k) \\ &= \tau \cdot \big(\tau \cdot s_2(d_k)\big) + s_2(d_k) \\ &= \tau \cdot \pi_1\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big). \end{aligned}$$

We find thus that equation (4) is valid for all $1 \leq j \leq s$. Now it will be evident from equation (2) that the same holds for $j = 0$; so we see that equation (3) is correct for $n = 1$. Now suppose that (3) is proved up to and including n. We will prove it for $n+1$. Choose $k \geq 1$. Again, let first of all $j = s$. Then we find

$$\begin{aligned} \pi_{n+1} \circ \tau_I \circ \partial_H(Z_s^k) &= s_2(d_k) \cdot \pi_n \circ \tau_I \circ \partial_H(Z_{s-1}^{k+1}) \\ &\overset{(3)}{=} s_2(d_k) \cdot \tau \cdot \pi_n\Big(\prod_{i=k+1}^{\infty} s_2(d_i)\Big) \\ &= \pi_{n+1}\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big). \end{aligned}$$

Suppose that for $1 < j \leq s$ we have already proved

$$\pi_{n+1} \circ \tau_I \partial_H(Z_j^k) = \tau \cdot \pi_{n+1}\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big), \tag{5}$$

then we find for $j-1$:

$$\begin{aligned}\pi_{n+1} \circ \tau_I \partial_H(Z_{j-1}^k) &= \tau \cdot \pi_{n+1} \circ \tau_I \partial_H(Z_j^k) + s_2(d_k) \cdot \pi_n \circ \tau_I \circ \partial_H(Z_{j-2}^{k+1}) \\ &\overset{(3)}{=} \tau \cdot \pi_{n+1} \circ \tau_I \partial_H(Z_j^k) + s_2(d_k) \cdot \tau \pi_n \Big(\prod_{i=k+1}^{\infty} s_2(d_i)\Big) \\ &\overset{(5)}{=} \tau \cdot \pi_{n+1}\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big).\end{aligned}$$

So for $1 \leq j \leq s$ we find that equation (5). Now it it trivial to see with the aid of equation (2) that equation (5) is valid for all $0 \leq j \leq s$. But this means that equation (3) is correct for $n+1$. So we find that this equation holds for all $n \geq 1$. Take, in particular, $k = 1$ and $j = 0$. Then we find for all $n \geq 1$:

$$\pi_n \circ \tau_I \circ \partial_H\Big(B_s \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big) = \tau \cdot \pi_n\Big(\prod_{i=1}^{\infty} s_2(d_i)\Big);$$

thus, using *AIP*, we find that theorem (4.2) is proved since $(d_i)_i \in D^\omega$ was arbitrary chosen.

Remarks (4.3)

First of all, we see that we did not use in the proof of theorem (4.2) any of the theory that we developed so far. The main reason for giving this proof is that it illustrates the problem that will occur if the capacity of the buffer tends to infinity: we see that while we are busy with the induction to n, we need a "small" reversed induction to j; if the capacity of the buffer becomes infinite—we have a queue with unbounded capacity—we can no longer make this reversed induction to j. Therefore, we will need another method to solve this problem.

Secondly, we also see that the above proof makes heavily use of axiom T2. If we have branching bisimulation (q.v. [**12**]), we can not just copy the above proof since in there we do not have T2. In that case we will actually need methods that will follow hereinafter (when we will show that a queue with unbounded capacity satisfies our correctness criterion) and then we can imitate the proof of (4.4) to obtain the result of theorem (4.2). This will be done in theorem (4.7).

We will give a guarded recursive specification of a queue Q over a data set D with input channel 1 and output channel 2 (see figure 5). We suppose that the data set contains more than one element. It is known that an infinity guarded recursive specification of Q can be given with the subsequent equations:

$$Q = Q_\lambda = \sum_{d \in D} r_1(d) \cdot Q_d, \tag{6}$$

$$Q_{d*\sigma} = s_2(d) \cdot Q_\sigma + \sum_{e \in D} r_1(e) \cdot Q_{d*\sigma*e}, \tag{7}$$

for any word $\sigma \in D^*$ and any $d \in D$. Here, we use $\lambda \in D^*$ for the empty word. The asterisk $(*)$ in equation (7) stands for the concatenation of words.

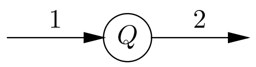

Figure 5. A queue Q with input channel 1 and output channel 2.

Below, we will prove that the queue is a correct communication protocol. But first, we will provide in figure 6 a picture of the left-hand side of equation (8), in which we can see that all the "lines" that go up in the graph are labeled with a τ.

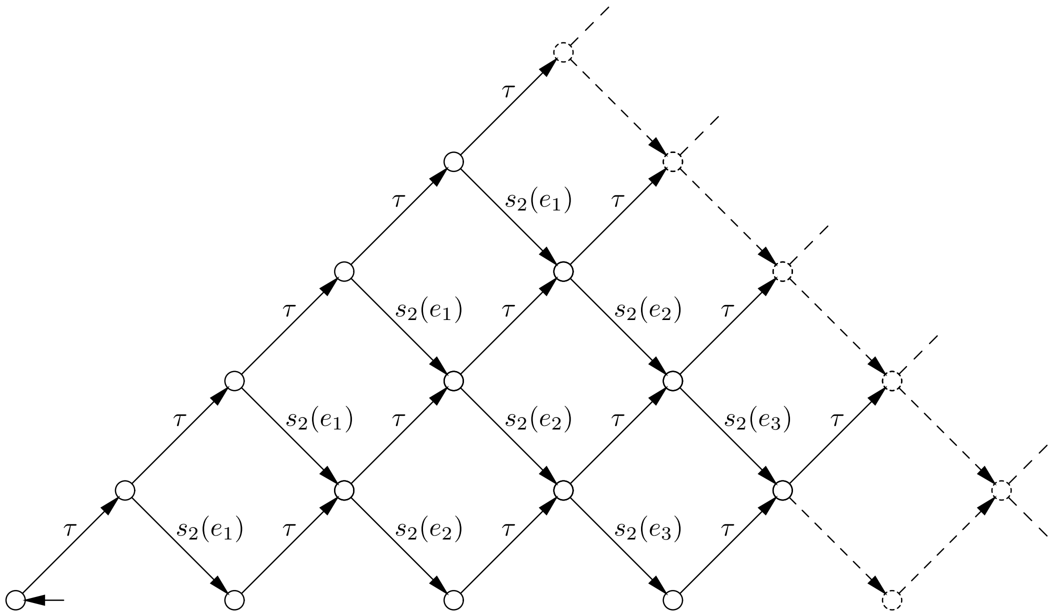

Figure 6. A graph of the left-hand side of equation (8)

Theorem (4.4)

The queue Q is a correct communication protocol.

Proof. Let $(e_i)_i \in D^\omega$. Let $H = \{r_1(d), s_1(d) : d \in D\}$ and $I = \{c_1(d) : d \in D\}$. In accord with definition (4.1), we are to show that

$$\tau_I \circ \partial_H\Big(Q_\lambda \parallel \prod_{i=1}^{\infty} s_1(e_i)\Big) = \tau \cdot \prod_{i=1}^{\infty} s_2(e_i). \tag{8}$$

We will show this by arguing with *GIP*. So let $J = \{s_2(d) : d \in D\}$. We will show that for all $n \geq 1$, for all $(e_i)_i \in D^\omega$ and for all $k \geq 1$ the following holds:

$$\pi_J^n \circ \tau_I \circ \partial_H\Big(Q_\lambda \parallel \prod_{i=1}^{\infty} s_1(e_i)\Big) = \tau \cdot \prod_{i=1}^{n} s_2(e_i), \tag{9}$$

$$\pi_J^n \circ \tau_I \circ \partial_H\Big(Q_{e_1 * \cdots * e_k} \parallel \prod_{i=k+1}^{\infty} s_1(e_i)\Big) = \tau \cdot \prod_{i=1}^{n} s_2(e_i). \tag{10}$$

We will prove equations (9) and (10) with induction on n. But first we will do some preliminary calculations. Therefore we will introduce the following abbreviations (for all $k \geq 1$):

$$Z_0 = Q_\lambda \parallel \prod_{i=1}^{\infty} s_1(e_i),$$

$$Z_k = Q_{e_1 * \cdots * e_k} \parallel \prod_{i=k+1}^{\infty} s_1(e_i).$$

Now we will calculate for all $k \geq 0 : \partial_H(Z_k)$. First we handle the case $k = 0$:

$$\partial_H(Z_0) = \partial_H\Big(Q_\lambda \parallel \prod_{i=1}^{\infty} s_1(e_i)\Big)$$

$$
\begin{aligned}
&= \partial_H\Big(Q_\lambda \mid \prod_{i=1}^{\infty} s_1(e_i)\Big)\\
&= c_1(e_1)\cdot \partial_H\Big(Q_{e_1} \parallel \prod_{i=2}^{\infty} s_1(e_i)\Big)\\
&= c_1(e_1)\cdot \partial_H(Z_1).
\end{aligned}
$$

Now we treat the case $k \geq 1$:

$$
\begin{aligned}
\partial_H(Z_k) &= \partial_H\Big(Q_{e_1*\cdots*e_k} \parallel \prod_{i=k+1}^{\infty} s_1(e_i)\Big)\\
&= \partial_H\Big(Q_{e_1*\cdots*e_k} \mathbin{\|\!\_} \prod_{i=k+1}^{\infty} s_1(e_i)\Big)\\
&\quad + \partial_H\Big(Q_{e_1*\cdots*e_k} \mid \prod_{i=k+1}^{\infty} s_1(e_i)\Big)\\
&= s_2(e_1)\cdot \partial_H\Big(Q_{e_2*\cdots*e_k} \parallel \prod_{i=k+1}^{\infty} s_1(e_i)\Big)\\
&\quad + c_1(e_{k+1})\cdot \partial_H\Big(Q_{e_1*\cdots*e_{k+1}} \mid \prod_{i=k+2}^{\infty} s_1(e_i)\Big)\\
&= s_2(e_1)\cdot \partial_H\Big(Q_{e_2*\cdots*e_k} \parallel \prod_{i=k+1}^{\infty} s_1(e_i)\Big) + c_1(e_{k+1})\cdot \partial_H(Z_{k+1})
\end{aligned}
$$

Now we will define an auxiliary linear unary operator as follows: Let $\{b, t\}$ be a set of function names and fix an arbitrary $i \in I$. Consider the linear functional specification hereinafter:

$$
\begin{aligned}
E(b,t) = &\left\{\, b(a) = a : a \in A \setminus I \,\right\}\\
\cup &\left\{\, b(a) = i : a \in I \,\right\}\\
\cup &\left\{\, t(a) = a : a \in A \setminus I \,\right\}\\
\cup &\left\{\, t(a) = \tau : a \in I \,\right\}\\
\cup &\left\{\, b(a\cdot x) = b(a)\cdot b(x) : a \in A \setminus J \,\right\}\\
\cup &\left\{\, b(a\cdot x) = b(a)\cdot t(x) : a \in J \,\right\}\\
\cup &\left\{\, t(a\cdot x) = t(a)\cdot t(x) : a \in A \,\right\}.
\end{aligned}
$$

According to *ODP* there is a valuation $\varphi : \{b, t\} \longrightarrow F$ that solves the system $E(b, t)$. Let us say $\varphi(b) = \beta$ and $\varphi(t) = \gamma$. We can see from the defining equations for γ that this operator behaves just like τ_I. We will show that $\gamma = \tau_I$. Let n be a function name. Consider the following linear functional specification:

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

It will be clear that τ_I solves this system. But from the defining equations of γ it is also clear that γ solves this system, too. Thus we can conclude with the aid of *OSP* that $\gamma = \tau_I$. We find thus the following functional equation for all $x \in P$ and $a \in J$:

$$
\beta(a\cdot x) = \beta(a)\cdot \tau_I(x). \tag{11}
$$

In fact we see in this equation the crux of the operator β. We can see this operator as some sort of "booby-trap". Only if the dangerous atomic actions pass, this operator will "detonate" and it becomes the abstraction operator afterwards. In all other circumstances only the abstraction set will be reduced by replacing all the internal actions for the fixed $i \in I$. This technique is called pre-abstraction and is studied in [**2**]. In the body of the proof it will be fine to have this situation but in the end we will treat all the atomic actions equal. So we will have to defuse this booby-trap. We will show hereinafter that $\tau_I \circ \beta = \tau_I$. Let thereunto $\{r, s\}$ be a set of function names. Consider the following linear functional specification:

$$\begin{aligned} E(r,s) = &\left\{\, r(a) = s(a) = a : a \in A \setminus I \,\right\} \\ &\cup \left\{\, r(a) = s(a) = \tau : a \in I \,\right\} \\ &\cup \left\{\, r(a \cdot x) = r(a) \cdot r(x) : a \in A \setminus J \,\right\} \\ &\cup \left\{\, r(a \cdot x) = r(a) \cdot s(x) : a \in J \,\right\} \\ &\cup \left\{\, s(a \cdot x) = s(a) \cdot s(x) : a \in A \,\right\}. \end{aligned}$$

We will show that $(\tau_I \circ \beta, \tau_I)$ is a solution for this system. First we will handle the boundary conditions. We see that

$$\tau_I \circ \beta(a) = \tau_I(a),$$

since $\beta(a) = a$ for all $a \in A \setminus I$ and $\beta(a) = i$ for all $i \in I$. Thus the boundary conditions are satisfied. Now we will treat the functional equations. It is evident that the functional equations for s are satisfied for all atomic actions. So we will only show that the functional equations for r are valid. Let $a \notin J$. Then we find:

$$\begin{aligned} \tau_I \circ \beta(a \cdot x) &= \tau_I\big(\beta(a) \cdot \beta(x)\big) \\ &= \tau_I \circ \beta(a) \cdot \tau_I \circ \beta(x). \end{aligned}$$

Now let $a \in J$. Then we obtain with the aid of (11):

$$\begin{aligned} \tau_I \circ \beta(a \cdot x) &= \tau_I\big(\beta(a) \cdot \tau_I(x)\big) \\ &= \tau_I \circ \beta(a) \cdot \tau_I \circ \tau_I(x) \\ &= \tau_I \circ \beta(a) \cdot \tau_I(x). \end{aligned}$$

We used here the following fact: $\tau_I \circ \tau_I = \tau_I$. It is trivial to see that $\tau_I \circ \tau_I$ is also a solution for the linear functional specification $E(n)$ that we defined hereinbefore, so with the aid of *OSP* we find that $\tau_I \circ \tau_I = \tau_I$. We find thus that the functional equations are valid. If we take on the other hand $r = s = \tau_I$ then it is not difficult to see that this solves the system $E(r, s)$, too. So in accordance with *OSP* we may conclude that these solutions must be equal. In particular we find:

$$\tau_I \circ \beta = \tau_I. \tag{12}$$

At this point we will return to the calculations that we were busy with. Our aim is to calculate $\tau_I \circ \partial_H(Z_k)$. If we look at the so far deduced equations for $\partial_H(Z_k)$, we can see that if we apply the abstraction operator on them, we obtain unguarded recursive equations. Hence, this type of calculating is in a certain way too coarse. Therefore, we have introduced β. We can apply it on the equations and we still have guardedness since for all atomic actions $a \in A$ we have $\beta(a) = a$. In certain subterms where we actually want an abstraction operator, this β will transform into τ_I after passing the "right" atomic actions.

Now we will calculate for all $k \geq 0 : \beta \circ \partial_H(Z_k)$. Again we will make a case distinction. First, let $k = 0$. Then we find at once

$$\beta \circ \partial_H(Z_0) = c_1(e_1) \cdot \beta \circ \partial_H(Z_1).$$

We will handle the case $k \geq 1$. We will use (11) here.

$$\beta \circ \partial_H(Z_k) = s_2(e_1) \cdot \tau_I \circ \partial_H\Big(Q_{e_2 * \cdots * e_k} \parallel \prod_{i=k+1}^{\infty} s_1(e_i)\Big) + c_1(e_{k+1}) \cdot \beta \circ \partial_H(Z_{k+1}).$$

At this point we will calculate for all $n \geq 1$ and for all $k \geq 0$ the nth projection of $\beta \circ \partial_H(Z_k)$. First, we will consider the case $k = 0$.

$$\pi_J^n \circ \beta \circ \partial_H(Z_0) = c_1(e_1) \cdot \pi_J^n \circ \beta \circ \partial_H(Z_1)$$

Now we will treat the case $k \geq 1$. Here we will have two subcases: $n = 1$ and $n > 1$. First, let $n = 1$ and consider the following

$$\pi_J^1 \circ \beta \circ \partial_H(Z_k) = s_2(e_1) + c_1(e_{k+1}) \cdot \pi_J^1 \circ \beta \circ \partial_H(Z_{k+1}). \tag{13}$$

Now let $n \geq 1$. Then we calculate

$$\pi_J^{n+1} \circ \beta \circ \partial_H(Z_k) = s_2(e_1) \cdot \pi_J^n \circ \tau_I \circ \partial_H\Big(Q_{e_2 * \cdots * e_k} \parallel \prod_{i=k+1}^{\infty} s_1(e_i)\Big) + c_1(e_{k+1}) \cdot \pi_J^{n+1} \circ \beta \circ \partial_H(Z_{k+1}). \tag{14}$$

Now we have all the prerequisites needed to establish the induction mentioned in the beginning of the proof. Recall that we proposed to verify that equations (9) and (10) hold for all $n \geq 1$. Thus, we will verify the basis of our induction. Let $n = 1$. Consider the following guarded recursive specification E_1:

$$E_1 = \big\{\, X_k = s_2(e_1) + c_1(e_{k+1}) \cdot X_{k+1} \mid k \geq 1 \,\big\}.$$

With the aid of equation (13) we see immediately that, if we put for all $k \geq 1$:

$$X_k = \pi_J^1 \circ \beta \circ \partial_H(Z_k),$$

this solves the guarded recursive specification E_1. Now consider the guarded recursive specification F_1:

$$F_1 = \big\{\, X = s_2(e_1) + c_1(e_{k+1}) \cdot X \,\big\}.$$

According to *RDP* there is a solution for this system. Let us say that x is a solution for F_1. If we put $X_k = x$ for all $k \geq 1$, we see that x is also a solution for the guarded recursive specification E_1. In accordance with *RSP* we may conclude now that for all $k \geq 1$:

$$x = \pi_J^1 \circ \beta \circ \partial_H(Z_k).$$

Since x is the solution for F_1, we see that the following equation is valid:

$$x = s_2(e_1) + c_1(e_{k+1}) \cdot x.$$

The guard $c_1(e_{k+1})$ is an internal atomic action. This means that we may apply *KFAR* on this equation. This yields: $\tau_I(x) = \tau \cdot s_2(e_1)$. So we find

$$\begin{aligned} \tau \cdot s_2(e_1) &= \tau_I(x) \\ &= \tau_I \circ \pi_J^1 \circ \beta \circ \partial_H(Z_k) \\ &= \pi_J^1 \circ \tau_I \circ \beta \circ \partial_H(Z_k) && \text{because of (3.12)} \\ &= \pi_J^1 \circ \tau_I \circ \partial_H(Z_k). && \text{because of (12)} \end{aligned}$$

Thus, we see that equation (10) holds for $n = 1$. Now we will show that equation (9) is correct for $n = 1$. Let thereunto $k = 0$ and consider the subsequent display.

$$\begin{aligned}\tau_I \circ \pi^1_J \circ \beta \circ \partial_H(Z_0) &= \tau \cdot \tau_I \circ \pi^1_J \circ \beta \circ \partial_H(Z_1)\\ &= \tau \cdot \pi^1_J \circ \tau_I \circ \partial_H(Z_1)\\ &= \tau \cdot \tau \cdot s_2(e_1)\\ &= \tau \cdot s_2(e_1),\end{aligned}$$

since we just have seen above that for all $k \geq 1 : \pi^1_J \circ \tau_I \circ \partial_H(Z_k) = \tau \cdot s_2(e_1)$. Thus, we see that both equations (9) and (10) are proved for $n = 1$. Now let us assume that these equations are valid for all $(e_i)_i \in D^\omega$ and $k \geq 1$ for a certain $n \geq 1$. Then we will prove these equations for $n+1$. This proof will be more or less the same as the verification of the case $n = 1$. We will work out equation (14) a bit more. Let for all $k \geq 1 : e'_i = e_{i+1}$ (we will use the induction hypothesis for $(e'_i)_i \in D^\omega$ and $k - 1 \geq 0$). Then we find

$$\begin{aligned}\pi^{n+1}_J \circ \beta \circ \partial_H(Z_k) &= s_2(e_1) \cdot \pi^n_J \circ \tau_I \circ \partial_H\Big(Q_{e'_1 * \cdots * e'_{k-1}} \parallel \prod_{i=k}^{\infty} s_1(e'_i)\Big)\\ &\quad + c_1(e_{k+1}) \cdot \pi^{n+1}_J \circ \beta \circ \partial_H(Z_{k+1})\\ &= s_2(e_1) \cdot \tau \cdot \prod_{i=1}^{n} s_1(e'_i) + c_1(e_{k+1}) \cdot \pi^{n+1}_J \circ \beta \circ \partial_H(Z_{k+1})\\ &= \prod_{i=1}^{n+1} s_2(e_i) + c_1(e_{k+1}) \cdot \pi^{n+1}_J \circ \beta \circ \partial_H(Z_{k+1}). \end{aligned} \tag{15}$$

Now consider the following guarded recursive specification E_2:

$$E_2 = \Big\{ Y_k = \prod_{i=1}^{n+1} s_2(e_i) + c_1(e_{k+1}) \cdot Y_{k+1} \;\Big|\; k \geq 1 \Big\}.$$

We see with the aid of equation (15), if we put for all $k \geq 1$

$$Y_k = \pi^{n+1}_J \circ \beta \circ \partial_H(Z_k)$$

that this is a solution for the guarded recursive specification E_2. Now consider the guarded recursive specification F_2:

$$F_2 = \Big\{ Y = \prod_{i=1}^{n+1} s_2(e_i) + c_1(e_{k+1}) \cdot Y \;\Big|\; k \geq 1 \Big\}.$$

With the aid of *RDP* we know that this system has a solution. Let us say that y is the solution for F_2. If we put for all $k \geq 1 : Y_k = y$, then we see that this also solves the guarded recursive specification E_2. So with the aid of *RSP* we conclude that for all $k \geq 1$:

$$y = \pi^{n+1}_J \circ \beta \circ \partial_H(Z_k).$$

Observe that the following equation holds

$$y = c_1(e_{k+1}) \cdot y + \prod_{i=1}^{n+1} s_2(e_i),$$

so we may apply *KFAR* since $c_1(e_{k+1}) \in I$. We find thus that

$$\tau_I(y) = \tau \cdot \prod_{i=1}^{n+1} s_2(e_i).$$

On the other hand, we find for all $k \geq 1$

$$\begin{aligned} \tau_I(y) &= \tau_I \circ \pi_J^{n+1} \circ \beta \circ \partial_H(Z_k) \\ &= \pi_J^{n+1} \circ \tau_I \circ \beta \circ \partial_H(Z_k) && \text{because of (3.12)} \\ &= \pi_J^{n+1} \circ \tau_I \circ \partial_H(Z_k). && \text{because of (12)} \end{aligned}$$

Combining these two inferences, we see that equation (10) is valid for $n+1$. Now we will handle the case $k = 0$. Since

$$\pi_J^{n+1} \circ \tau_I \circ \partial_H(Z_0) = \tau \cdot \pi_J^{n+1} \circ \tau_I \circ \partial_H(Z_1),$$

we find, using the just derived results for $k = 1$, that

$$\pi_J^{n+1} \circ \tau_I \circ \partial_H(Z_0) = \tau \cdot \tau \cdot \prod_{i=1}^{n+1} s_2(e_i) = \tau \cdot \prod_{i=1}^{n+1} s_2(e_i).$$

This means that equation (9) is also valid. This will end proof of equations (9) and (10). Now we find in particular for all $n \geq 1$ and $k = 0$ that equation (9) is valid. If we adjust this equation a little as is done below

$$\pi_J^n \circ \tau_I \circ \partial_H\Big(Q_\lambda \parallel \prod_{i=1}^{\infty} s_1(e_i)\Big) = \pi_J^n\Big(\tau \cdot \prod_{i=1}^{\infty} s_2(e_i)\Big),$$

we may apply *GIP* and we find

$$\tau_I \circ \partial_H\Big(Q_\lambda \parallel \prod_{i=1}^{\infty} s_1(e_i)\Big) = \tau \cdot \prod_{i=1}^{\infty} s_2(e_i).$$

According to definition (4.1), we may conclude that Q_λ is a correct communication protocol. This will end the proof of (4.4).

Remark (4.5)

As we can see in the proof of the former theorem, we did not use axiom T2. In fact, we only used axiom T1, which means that this proof can also be given if we had used branching τ-laws.

We will formulate a lemma on β for later reference.

Lemma (4.6) *The selective abstraction lemma*

Let $S, I \subseteq A$. Then there is a linear unary operator $\beta(I,S) \in F$ with the following properties:

(i) $\beta(I,S)(a) = a \quad (\forall a \in A)$

(ii) $\beta(I,S)(a \cdot x) = a \cdot \beta(I,S)(x) \quad (\forall a \in A \setminus S, \forall x \in P)$

(iii) $\beta(I,S)(a \cdot x) = a \cdot \tau_I(x) \quad (\forall a \in S, \forall x \in P)$

(iv) $\tau_I \circ \beta(I,S) = \tau_I$

Proof. Replace in the proof of theorem (4.4) the set J for the set S and do not rename internal actions to a specific one. Then the introduction of the linear unary operator β will satisfy the requirements of (4.6). And we can take $\beta(I,S) = \beta$. This will end the proof of our lemma.

We will supply another proof of theorem (4.2) in which we will not use axiom T2. We will use the method that we established in the proof of theorem (4.4).

Theorem (4.7)

Let B_s be the s-bit buffer of page 17. For all $(d_i)_i \in D^\omega$ the following holds:

$$\tau_I \circ \partial_H\Big(B_s \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big) = \tau \cdot \prod_{i=1}^{\infty} s_2(d_i).$$

Proof. We will (tacitly) use the notations as they appear in the former theorems. It is very easy to see that the following equations are valid ($1 \le j < s$).

$$\begin{aligned}
\partial_H(Z_0^k) &= c_1(d_k) \cdot \partial_H(Z_1^k)\\
\partial_H(Z_j^k) &= c_1(d_{k+j}) \cdot \partial_H(Z_{j+1}^k) + s_2(d_k) \cdot \partial_H(Z_{j-1}^{k+1})\\
\partial_H(Z_s^k) &= s_2(d_k) \cdot \partial_H(Z_{s-1}^{k+1}).
\end{aligned}$$

In contrast to the method used in theorem (4.2), we will follow here the method of theorem (4.4). So let $J = \{s_2(d) : d \in D\}$ and $\beta = \beta(I, J)$; see lemma (4.6). Thus, we find using equation (11)

$$\begin{aligned}
\beta \circ \partial_H(Z_0^k) &= c_1(d_k) \cdot \beta \circ \partial_H(Z_1^k)\\
\beta \circ \partial_H(Z_j^k) &= c_1(d_{k+j}) \cdot \beta \circ \partial_H(Z_{j+1}^k) + s_2(d_k) \cdot \tau_I \circ \partial_H(Z_{j-1}^{k+1})\\
\beta \circ \partial_H(Z_s^k) &= s_2(d_k) \cdot \tau_I \circ \partial_H(Z_{s-1}^{k+1}).
\end{aligned}$$

We will prove the following claim with induction to n.

$$(16)\qquad \forall k, n \ge 1,\ \forall 0 \le j \le s : \pi_J^n \circ \tau_I \circ \partial_H(Z_j^k) = \begin{cases} \tau \cdot \pi_J^n\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big), & \text{if } 0 \le j < s;\\ \pi_J^n\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big), & \text{if } j = s.\end{cases}$$

First, we will prove equation (16) for $n = 1$. For $j = s$ we immediately see that this is the case. Now assume that $1 \le j < s$. We will calculate

$$\pi_J^1 \circ \beta \circ \partial_H(Z_j^k) = c_1(d_{k+j}) \cdot \pi_J^1 \circ \beta \circ \partial_H(Z_{j+1}^k) + s_2(d_k).$$

Now consider the following guarded recursive specification E_1.

$$E_1 = \big\{\, X_j = c_1(d_{k+j}) \cdot X_{j+1} + s_2(d_k),\ X_s = s_2(d_k) \mid 1 \le j < s \,\big\}$$

It will be clear that if we put $X_j = \pi_J^1 \circ \beta \circ \partial_H(Z_j^k)$ for all $1 \le j \le s$, this will solve the system E_1. Now contemplate the guarded recursive specification F_1:

$$F_1 = \big\{\, X = c_1(d_{k+j}) \cdot X + s_2(d_k) \,\big\}$$

and let x be a solution for this system accordingly to *RDP*. If we put for all $1 \le j < s : X_j = x$, we see that this will solve the system E_1, too. So with the help of *RSP* we may conclude for all $1 \le j < s$:

$$x = \pi_J^1 \circ \beta \circ \partial_H(Z_j^k).$$

Now we will apply *KFAR* on the equation $x = c_1(d_{k+j}) \cdot x + s_2(d_k)$ and we find thus $\tau_I(x) = \tau \cdot s_2(d_k)$. It is easy to see that, using equation (12), the following holds:

$$\begin{aligned}
\tau \cdot s_2(d_k) &= \tau_I(x)\\
&= \tau_I \circ \pi_J^1 \circ \beta \circ \partial_H(Z_j^k)\\
&= \pi_J^1 \circ \tau_I \circ \beta \circ \partial_H(Z_j^k)\\
&= \pi_J^1 \circ \tau_I \circ \partial_H(Z_j^k).
\end{aligned}$$

So we find that the claim holds for $1 \le j \le s$. It is trivial to see that it is also valid for $j = 0$. Thus we see that equation (16) is correct for $n = 1$. Suppose that this claim holds for a certain $n \ge 1$ then we prove it for $n+1$. First we take $j = s$:

$$\begin{aligned}\pi_J^{n+1} \circ \tau_I \circ \partial_H(Z_s^k) &= s_2(d_k) \cdot \pi_J^n \circ \tau_I \circ \partial_H(Z_{s-1}^{k+1}) \\ &\overset{(16)}{=} s_2(d_k) \cdot \tau \cdot \pi_J^n\Big(\prod_{i=k+1}^{\infty} s_2(d_i)\Big) \\ &= \pi_J^{n+1}\Big(\prod_{i=k}^{\infty} s_2(d_i)\Big).\end{aligned}$$

Now we take $1 \le j < s$. Consider the following guarded recursive specification E_2:

$$E_2 = \Big\{\, Y_j = c_1(d_{k+j}) \cdot Y_{j+1} + \prod_{i=k}^{k+n+1} s_2(d_i), Y_s = \prod_{i=k}^{k+n+1} s_2(d_i) \mid 1 \le j < s \,\Big\}.$$

We see at once that, if we put $Y_j = \pi_J^{n+1} \circ \beta \circ \partial_H(Z_j^k)$ for all $1 \le j < s$, this solves the system E_2 using the induction hypothesis. Now consider also the guarded recursive specification F_2

$$F_2 = \Big\{\, Y = c_1(d_{k+j}) \cdot Y + \prod_{i=k}^{k+n+1} s_2(d_i) \,\Big\}.$$

With the aid of *RDP* we now that this guarded recursive specification has a solution, say y. If we put for all $1 \le j < s : Y_j = y$ then we see that this will solve the system E_2. But then we find with *RSP* that

$$y = \pi_J^{n+1} \circ \beta \circ \partial_H(Z_j^k).$$

If we apply *KFAR* on the equation

$$y = c_1(d_{k+j}) \cdot y + \prod_{i=k}^{k+n+1} s_2(d_i),$$

we find that

$$\tau_I(y) = \tau \cdot \prod_{i=k}^{k+n+1} s_2(d_i).$$

So we can deduce:

$$\begin{aligned}\tau \cdot \prod_{i=k}^{k+n+1} s_2(d_i) &= \tau_I \circ \pi_J^{n+1} \circ \beta \circ \partial_H(Z_j^k) \\ &= \pi_J^{n+1} \circ \tau_I \circ \beta \circ \partial_H(Z_j^k) \\ &= \pi_J^{n+1} \circ \tau_I \circ \partial_H(Z_j^k).\end{aligned}$$

So we see that equation (16) is correct for $1 \le j \le s$. Now it is trivial to see that the same yields for $j = 0$. This concludes the induction step and we find, in particular, for $k = 1$ and $j = 0$ that for all $n \ge 1$:

$$\pi_J^n \circ \tau_I \circ \partial_H\Big(B_s \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big) = \tau \cdot \pi_J^n\Big(\prod_{i=1}^{\infty} s_2(d_i)\Big).$$

Thus, using *GIP*, we finish the proof of (4.7).

Next, we will give an example of an alternating bit protocol with a time out. See figure 7 hereinafter.

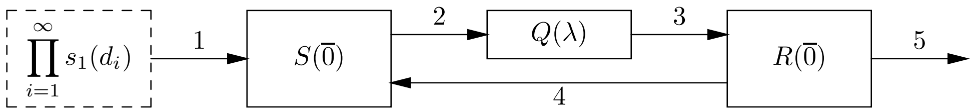

Figure 7. An alternating bit protocol with a time out.

The dashed box in figure 7 is not part of the protocol. It is a test row. Below, we will give the specification of the components that are present in figure 7. But first, we will explain some of the symbols used in the specification. The symbol $\overline{k}$ is common in group theory. It stands for

$$\overline{k} \equiv k \bmod 2 \in \mathbb{Z}/2\mathbb{Z}.$$

It is, in fact, the alternating bit. We use this notation since it will shorten notations somewhat. Furthermore, D is a finite set of datum elements. The abbreviation RA stands for read acknowledgement. The atomic action t stands for *time out* and it is this particular action that justifies the use of a queue with an unbounded capacity. The set E is the following:

$$E = \{d\overline{k} : d \in D, \overline{k} \in \mathbb{Z}/2\mathbb{Z}\}.$$

The abbreviation SM means send message and SA send acknowledgement.

$$\begin{aligned}
S(\overline{k}) &= \sum_{d \in D} r_1(d) \cdot S(d\overline{k}) \\
S(d\overline{k}) &= s_2(d\overline{k}) \cdot RA(d\overline{k}) \\
RA(d\overline{k}) &= r_4(\overline{k}) \cdot S(\overline{k+1}) + \big(r_4(\overline{k+1}) + t\big) \cdot S(d\overline{k}) \\
Q(\lambda) &= \sum_{x \in E} r_2(x) \cdot Q(x) \\
Q(x * \sigma) &= \sum_{y \in E} r_2(y) \cdot Q(x * \sigma * y) + s_3(x) \cdot Q(\sigma) \\
R(\overline{k}) &= \sum_{d \in D} r_3(d\overline{k}) \cdot SM(d\overline{k}) + \sum_{d \in D} r_3\big(d(\overline{k+1})\big) \cdot SA(\overline{k+1}) \\
SM(d\overline{k}) &= s_5(d) \cdot SA(\overline{k}) \\
SA(\overline{k}) &= s_4(\overline{k}) \cdot R(\overline{k+1})
\end{aligned}$$

Below, we will enumerate the encapsulation set H and the abstraction set I.

$$\begin{aligned}
H = &\; \big\{\, r_1(d), s_1(d) : d \in D \big\} \\
&\cup \big\{\, r_2(x), r_3(x), s_2(x), s_3(x) : x \in E \big\} \\
&\cup \big\{\, r_4(\overline{k}), s_4(\overline{k}) : \overline{k} \in \mathbb{Z}/2\mathbb{Z} \,\big\} \\
I = &\; \big\{\, c_1(d) : d \in D \,\big\} \\
&\cup \big\{\, c_2(x), c_3(x) : x \in E \,\big\} \\
&\cup \big\{\, c_4(\overline{k}) : \overline{k} \in \mathbb{Z}/2\mathbb{Z} \,\big\} \cup \{t\}.
\end{aligned}$$

Theorem (4.8)

$$\tau_I \circ \partial_H \Big(S(\overline{0}) \parallel Q(\lambda) \parallel R(\overline{0}) \parallel \prod_{i=1}^{\infty} s_1(d_i) \Big) = \tau \cdot \prod_{i=1}^{\infty} s_5(d_i).$$

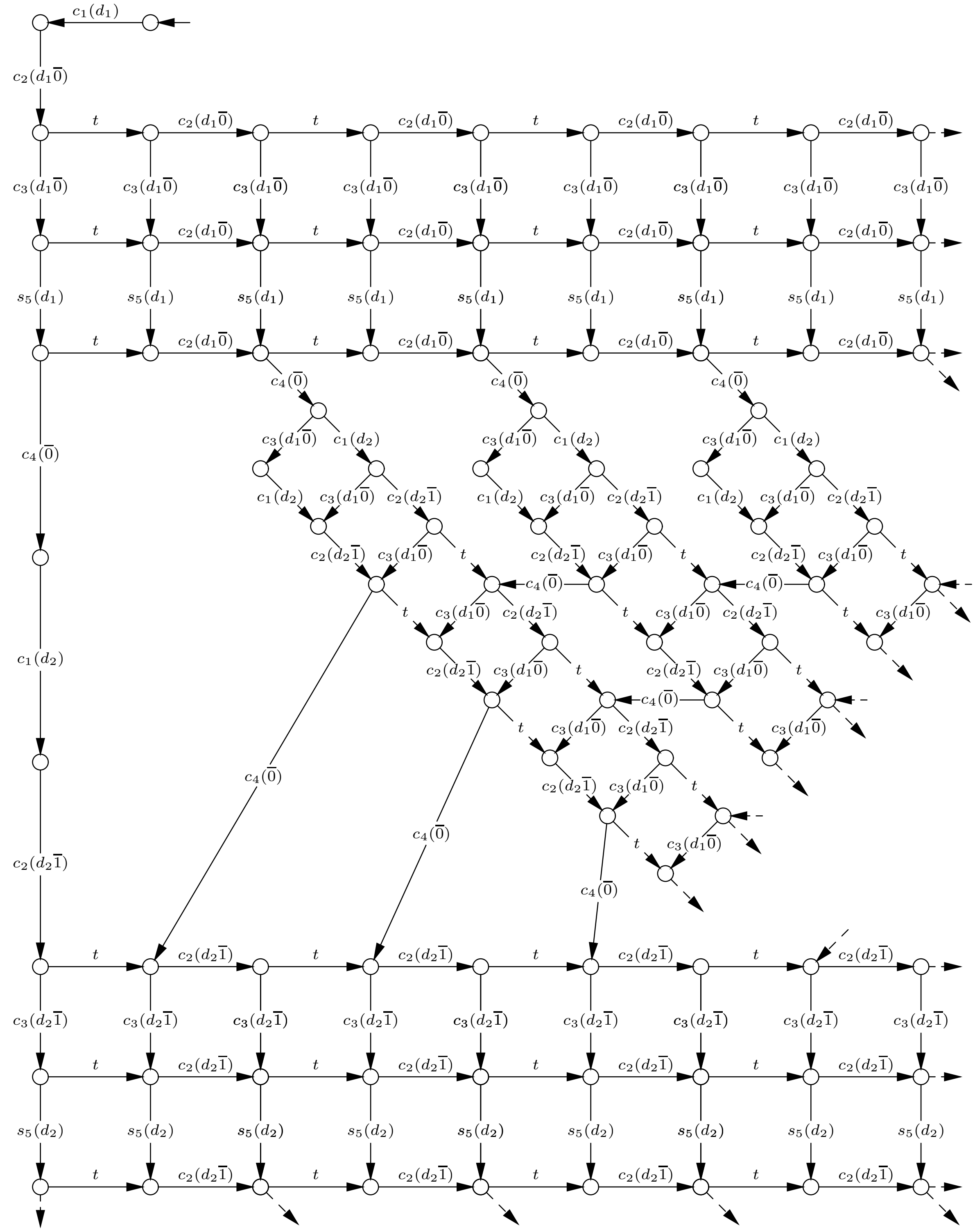

Figure 8. A beginning of a graph of $\partial_H\Big(S(\overline{0}) \parallel Q(\lambda) \parallel R(\overline{0}) \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big)$.

Proof. We will first calculate

$$\partial_H\Big(S(\overline{0}) \parallel Q(\lambda) \parallel R(\overline{0}) \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big).$$

Figure 8 (see page 29) is a picture of this process. We will introduce some notations for convenience sake. We will abbreviate $\tilde{d}_{k+1} = d_{k+1}\overline{k}$ and for all $n \geq 1$:

$$\tilde{d}_{k+1}^{n} = \underbrace{\tilde{d}_{k+1} * \cdots * \tilde{d}_{k+1}}_{n \text{ times}}.$$

If $n = 0$ we will use the convention $x^0 = \lambda$ for all words $x \in E^*$. Let for all $k \geq 0$ and any word $\sigma \in E^*$ the following be given:

$$\begin{aligned}
A_k(\sigma) &= S(\overline{k}) \parallel Q(\sigma) \parallel R(\overline{k}) \parallel \prod_{i=k+1}^{\infty} s_1(d_i)\\
B_k(\sigma) &= S(\tilde{d}_{k+1}) \parallel Q(\sigma) \parallel R(\overline{k}) \parallel \prod_{i=k+2}^{\infty} s_1(d_i)\\
C_k(\sigma) &= RA(\tilde{d}_{k+1}) \parallel Q(\sigma) \parallel R(\overline{k}) \parallel \prod_{i=k+2}^{\infty} s_1(d_i)\\
D_k(\sigma) &= RA(\tilde{d}_{k+1}) \parallel Q(\sigma) \parallel SM(\tilde{d}_{k+1}) \parallel \prod_{i=k+2}^{\infty} s_1(d_i)\\
E_k(\sigma) &= S(\tilde{d}_{k+1}) \parallel Q(\sigma) \parallel SM(\tilde{d}_{k+1}) \parallel \prod_{i=k+2}^{\infty} s_1(d_i)\\
F_k(\sigma) &= RA(\tilde{d}_{k+1}) \parallel Q(\sigma) \parallel SA(\overline{k}) \parallel \prod_{i=k+2}^{\infty} s_1(d_i)\\
G_k(\sigma) &= RA(\tilde{d}_{k+2}) \parallel Q(\sigma) \parallel SA(\overline{k}) \parallel \prod_{i=k+3}^{\infty} s_1(d_i).
\end{aligned}$$

Observe, that we are to evaluate $\partial_H\big(A_0(\lambda)\big)$. If we do so, we will find that there is much more needed. For, we will have to calculate $\partial_H\big(A_k(\lambda)\big)$ for all $k \geq 0$. If we make this calculation, we will see that we will also need to evaluate for all $k, n \geq 0$

$$\partial_H\big(A_{k+1}(\tilde{d}_{k+1}^{n})\big).$$

If we calculate this "one" we will find that this is enough to obtain a guarded recursive specification for $\partial_H\big(A_0(\lambda)\big)$ with the abbreviations that we can find above. First, we will give the result of the calculation. Second, with the aid of the guarded recursive specification, consisting of equations (17)–(27) and figure 8 of page 29, we will provide some explanation such that it will be not that hard for the reader to verify the result. For all $k \geq 0$ we have the following guarded recursive specification for $\partial_H\big(A_0(\lambda)\big)$.

$$\begin{aligned}
\partial_H\big(A_k(\lambda)\big) &= c_1(d_{k+1}) \cdot \partial_H\big(B_k(\lambda)\big) && (17)\\
\partial_H\big(A_{k+1}(\tilde{d}_{k+1}^{n})\big) &= c_1(d_{k+2}) \cdot \partial_H\big(B_{k+1}(\tilde{d}_{k+1}^{n})\big) && (n \geq 1)\\
&\quad + c_3(\tilde{d}_{k+1}) \cdot c_1(d_{k+2}) \cdot c_2(\tilde{d}_{k+2}) \cdot \partial_H\big(G_k(\tilde{d}_{k+1}^{n-1} * \tilde{d}_{k+2})\big) && (18)
\end{aligned}$$

$$\begin{aligned}
\partial_H\big(B_k(\lambda)\big) &= c_2(\tilde{d}_{k+1})\cdot\partial_H\big(C_k(\tilde{d}_{k+1})\big) && (19)\\
\partial_H\big(B_k(\tilde{d}^n_{k+1})\big) &= c_2(\tilde{d}_{k+1})\cdot\partial_H\big(C_k(\tilde{d}^{n+1}_{k+1})\big) && (n\geq 1)\\
&\quad + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(E_k(\tilde{d}^{n-1}_{k+1})\big) && (20)\\
\partial_H\big(B_{k+1}(\tilde{d}^n_{k+1}*\tilde{d}^m_{k+2})\big) &= c_2(\tilde{d}_{k+2})\cdot\partial_H\big(C_{k+1}(\tilde{d}^n_{k+1}*\tilde{d}^{m+1}_{k+2})\big) && (n\geq 1, m\geq 0)\\
&\quad + c_3(\tilde{d}_{k+1})\cdot c_2(\tilde{d}_{k+2})\cdot\partial_H\big(G_k(\tilde{d}^{n-1}_{k+1}*\tilde{d}^{m+1}_{k+2})\big) && (21)
\end{aligned}$$

$$\begin{aligned}
\partial_H\big(C_k(\tilde{d}^n_{k+1})\big) &= t\cdot\partial_H\big(B_k(\tilde{d}^n_{k+1})\big) && (n\geq 1)\\
&\quad + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(D_k(\tilde{d}^{n-1}_{k+1})\big) && (22)\\
\partial_H\big(C_{k+1}(\tilde{d}^n_{k+1}*\tilde{d}^m_{k+2})\big) &= t\cdot\partial_H\big(B_{k+1}(\tilde{d}^n_{k+1}*\tilde{d}^m_{k+2})\big) && (n,m\geq 1)\\
&\quad + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(G_k(\tilde{d}^{n-1}_{k+1}*\tilde{d}^m_{k+2})\big) && (23)
\end{aligned}$$

$$\begin{aligned}
\partial_H\big(D_k(\tilde{d}^n_{k+1})\big) &= t\cdot\partial_H\big(E_k(\tilde{d}^n_{k+1})\big) && (n\geq 0)\\
&\quad + s_5(d_{k+1})\cdot\partial_H\big(F_k(\tilde{d}^n_{k+1})\big) && (24)
\end{aligned}$$

$$\begin{aligned}
\partial_H\big(E_k(\tilde{d}^n_{k+1})\big) &= c_2(\tilde{d}_{k+1})\cdot\partial_H\big(D_k(\tilde{d}^{n+1}_{k+1})\big) && (n\geq 0)\\
&\quad + s_5(d_{k+1})\cdot c_2(\tilde{d}_{k+1})\cdot\partial_H\big(F_k(\tilde{d}^{n+1}_{k+1})\big) && (25)
\end{aligned}$$

$$\begin{aligned}
\partial_H\big(F_k(\tilde{d}^n_{k+1})\big) &= t\cdot c_2(\tilde{d}_{k+1})\cdot\partial_H\big(F_k(\tilde{d}^{n+1}_{k+1})\big) && (n\geq 0)\\
&\quad + c_4(\overline{k})\cdot\partial_H\big(A_{k+1}(\tilde{d}^n_{k+1})\big) && (26)
\end{aligned}$$

$$\begin{aligned}
\partial_H\big(G_k(\tilde{d}^n_{k+1}*\tilde{d}^m_{k+2})\big) &= t\cdot c_2(\tilde{d}_{k+2})\cdot\partial_H\big(G_k(\tilde{d}^n_{k+1}*\tilde{d}^{m+1}_{k+2})\big) && (n\geq 0, m\geq 1)\\
&\quad + c_4(\overline{k})\cdot\partial_H\big(B_{k+1}(\tilde{d}^n_{k+1}*\tilde{d}^m_{k+2})\big) && (27)
\end{aligned}$$

At this point we will explain this big guarded recursive specification with the aid of the figure. We will begin in the first node. At that place there is only one possible action that can be performed: the first element will be taken from the test row (we will go to the left in the graph). This corresponds to equation (17). Still there is one possible action: the test datum will be loaded in the queue (we will go down one node in the graph). This transition corresponds to equation (19). Now we entered the third node and here we have two possible actions: the queue can communicate with the receiver and the just read datum will be removed from the queue (we go one node down in the graph) or a time out can be given (we go to the right in the graph). We will call the first infinite horizontal trace the single BC-trace since in this phase the queue can only contain one type of data. We will discuss this trace. We can see that after the time out the queue can be loaded (we will go to the right) or the single BC-trace can be escaped by communicating with the receiver (we will go one node down in the graph). This behaviour is reiterated every two nodes in the single BC-trace. The equations that correspond to this trace are (20) and (22). As we said we can leave the single BC-trace with a communication (in the graph all the down going arrows are labelled with this action) and we will enter what we will call a DE-trace. We will discuss this trace. In fact, this trace is the same as the single BC-trace except that the down going arrows are labelled with a send action and then we will enter an F-trace. The equations that correspond with the DE-trace are (24) and (25). This F-trace consist of alternating time outs and loadings of the queue. The equation that accompanies this is (26). In the picture we can see that there are two different ways to escape the F-trace. The first possible arrow down connects the F-trace “directly” to the next single BC-trace. Then the pattern will that we already described will be repeated. We will explain the arrows that go southeastwards from the F-trace. As we can see in equation (26) these A-nodes contain old data: the acknowledgement has been performed but due to time outs there are still old datum elements in the queue. From these A-nodes we can take the next element of the test row (we go to the southeast in the graph) and we will enter a double BC-trace or we can communicate with the receiver (we go southwestwardly) and we will enter a G-trace. This corresponds to equation (18). We will discuss the double BC-traces. Now there are old data in the queue and moreover, new data can be added if we stay in this trace: alternating time outs and loading of new data will be performed (we go southeastwards in the graph). As with the single BC-traces we can escape the double one by communicating with the receiver (we go southwestwards from a node in the double BC-trace). This corresponds to the equations (21) and (23). Now we will discuss the G-traces. In this trace we can execute alternating time outs and loadings of the queue (we go southeastwards in the graph). We can escape a G-trace by the execution of an acknowledgement and we enter a double BC-trace with one old datum element less (we go to the left in the graph) or if we are already at the leftmost G-trace the next single BC-trace will be entered (the three long arrows in the graph that connect the leftmost G-trace with the second single BC-trace). The equation that corresponds to the G-traces is (27).

Now that it is plausible how this big guarded recursive specification can be obtained, we will discuss the structure of the sequel of the proof of this theorem. First we will make an assumption on a DE-trace for one fixed $p \geq 1$ (which is valid for $p = 1$). Then we will derive from this assumption a conclusion on an F-trace. To obtain this result we will need an intermediary result on the double BC-traces and the G-traces. This will be deduced first from the assumptions on the DE-trace. As soon as we verified all this we will show that the conclusion on the F-trace itself is a true statement. With this result we will prove that the same yields for the assumptions that we made. In fact, we set off a chain reaction: the assumptions for $p = 1$ are satisfied. This implies that the conclusions are valid for $p = 2$. But with this we can show that the assumptions for $p = 2$ are satisfied so the conclusion is valid for $p = 3$, et cetera. We will do this with induction on p. Once we have all these facts we will deduce a result on the single BC-traces and then a small calculation will finish the proof.

Let $J = \{s_5(d) : d \in D\}$. Let $\beta_1 = \beta(I, J)$; see for such operators and their properties lemma (4.6). We will use these properties tacitly throughout this verification. Let $p \geq 1$ be fixed. Suppose that for all $k, n \geq 0$ the following holds

$$\pi_J^p \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^n)\big) = t \cdot \pi_J^p \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) + \prod_{i=1}^{p} s_5(d_{k+i}) \tag{28}$$

$$\pi_J^p \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) = c_2(\tilde{d}_{k+1}) \cdot \partial_H\big(D_k(\tilde{d}_{k+1}^{n+1})\big) + \prod_{i=1}^{p} s_5(d_{k+i}) \tag{29}$$

Then we will prove that for all $k, n \geq 0$

$$\pi_J^p \circ \tau_I \circ \partial_H\big(F_k(\tilde{d}_{k+1}^n)\big) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \tag{30}$$

In order to prove this we will need an intermediary result that we will verify first. For all $k, l, m \geq 0$ we have

$$\left.\begin{array}{l} \pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+1}^l * \tilde{d}_{k+2}^m)\big) \\ \pi_J^p \circ \tau_I \circ \partial_H\big(G_k(\tilde{d}_{k+1}^l * \tilde{d}_{k+2}^{m+1})\big) \end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \tag{31}$$

We will prove (31) with induction on l. So let $l = 0$ and fix an arbitrary $k \geq 0$. Consider the following guarded recursive specification E_1.

$$\begin{aligned} E_1 = \Big\{ X_n^1 &= t \cdot Y_n^1 + \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ Y_n^1 &= c_2(\tilde{d}_{k+2}) \cdot X_{n+1}^1 + \prod_{i=1}^{p} s_5(d_{k+1+i}) \;\Big|\; n \geq 0 \Big\}. \end{aligned}$$

Because of the assumptions that we made in equations (28) and (29) we see that if we put for all $n \geq 0$

$$\begin{aligned} X_n^1 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) \\ Y_n^1 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(E_{k+1}(\tilde{d}_{k+2}^n)\big), \end{aligned}$$

this is a solution for the guarded recursive specification E_1. The subsequent ratiocination will frequently occur in this proof. In fact, we already applied this technique in the former theorems. So it will be convenient to give it a name for later reference. We will baptize it "a collapsing-argument". Now consider the following guarded recursive specification F_1, which is "E_1 without n".

$$\begin{aligned} F_1 = \{\, X_1 &= t \cdot Y_1 + \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ Y_1 &= c_2(\tilde{d}_{k+2}) \cdot X_1 + \prod_{i=1}^{p} s_5(d_{k+1+i}) \,\}. \end{aligned}$$

According to *RDP* there is a solution for F_1, say (x_1, y_1). So we have two equations

$$\begin{aligned} x_1 &= t \cdot y_1 + \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ y_1 &= c_2(\tilde{d}_{k+2}) \cdot x_1 + \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

On these equations we may apply $KFAR_2$ (see definition (2.16)) and we find

$$\tau_I(x_1) = \tau_I(y_1) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{32}$$

It will be clear that, if we put for all $n \geq 0$

$$X_n^1 = x_1, \quad Y_n^1 = y_1,$$

this will solve the system E_1, too. So with the aid of *RSP* we find that for all $n \geq 0$

$$\begin{aligned} x_1 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) \\ y_1 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(E_{k+1}(\tilde{d}_{k+2}^n)\big). \end{aligned}$$

And we obtain using lemma (4.6), theorem (3.12) and equation (32) that

$$\left.\begin{aligned} &\pi_J^p \circ \tau_I \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) \\ &\pi_J^p \circ \tau_I \circ \partial_H\big(E_{k+1}(\tilde{d}_{k+2}^n)\big) \end{aligned}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{33}$$

We will show this for the first equation.

$$\begin{aligned} \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) &= \tau_I(x_1) \\ &= \tau_I \circ \pi_J^p \circ \beta_1 \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) \\ &= \pi_J^p \circ \tau_I \circ \beta_1 \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) && \text{because of (3.12)} \\ &= \pi_J^p \circ \tau_I \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) && \text{see lemma (4.6)} \end{aligned}$$

Such a calculation can be made for E_{k+1}, too. We find thus that (33) is correct.

Now consider the guarded recursive specification E_2

$$\begin{aligned} E_2 = \{\, X_m^2 &= c_2(\tilde{d}_{k+2}) \cdot Y_{m+1}^2 + c_3(\tilde{d}_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ Y_{m+1}^2 &= t \cdot X_{m+1}^2 + c_3(\tilde{d}_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \mid m \geq 0 \,\}. \end{aligned}$$

Let $\beta_2 = \beta\big(I, \{c_3(\tilde{d}_{k+2})\}\big)$; see lemma (4.6). It is easy to see with the aid of equations (20), (22) and (33) that if we put for all $m \geq 0$

$$\begin{aligned} X^2_m &= \pi^p_J \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}^m_{k+2})\big) \\ Y^2_{m+1} &= \pi^p_J \circ \beta_2 \circ \partial_H\big(C_{k+1}(\tilde{d}^{m+1}_{k+2})\big), \end{aligned}$$

this will solve the system E_2. First, we will show this for the first equation of E_2.

$$\begin{aligned} \pi^p_J \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}^m_{k+2})\big) &\overset{(20)}{=} c_2(\tilde{d}_{k+2}) \cdot \pi^p_J \circ \beta_2 \circ \partial_H\big(C_{k+1}(\tilde{d}^{m+1}_{k+2})\big) \\ &\quad + c_3(\tilde{d}_{k+2}) \cdot \pi^p_J \circ \tau_I \circ \partial_H\big(E_{k+1}(\tilde{d}^{m-1}_{k+2})\big) \\ &\overset{(33)}{=} c_2(\tilde{d}_{k+2}) \cdot \pi^p_J \circ \beta_2 \circ \partial_H\big(C_{k+1}(\tilde{d}^{m+1}_{k+2})\big) \\ &\quad + c_3(\tilde{d}_{k+2}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

Now we will treat the second equation.

$$\begin{aligned} Y^2_{m+1} &= \pi^p_J \circ \beta_2 \circ \partial_H\big(C_{k+1}(\tilde{d}^{m+1}_{k+2})\big) \\ &\overset{(22)}{=} t \cdot \pi^p_J \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}^{m+1}_{k+2})\big) \\ &\quad + c_3(\tilde{d}_{k+2}) \cdot \pi^p_J \circ \tau_I \circ \partial_H\big(D_{k+1}(\tilde{d}^m_{k+2})\big) \\ &\overset{(33)}{=} t \cdot X^2_{m+1} + c_3(\tilde{d}_{k+2}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

We see that this solves indeed the guarded recursive specification E_2. Now we are going to use a collapsing-argument as with E_1: we construct a guarded recursive specification F_2 without m.

$$\begin{aligned} F_2 = \Big\{\, X_2 &= c_2(\tilde{d}_{k+2}) \cdot Y_2 + c_3(\tilde{d}_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ Y_2 &= t \cdot X_2 + c_3(\tilde{d}_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \,\Big\}. \end{aligned}$$

With *RDP* we see that there is a solution (x_2, y_2) for F_2 that is also a solution for the system E_2. With *KFAR*$_2$ we find that

$$\tau_I(x_2) = \tau_I(y_2) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i})$$

and with *RSP* we see that for all $m \geq 0$

$$x_2 = \pi^p_J \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}^m_{k+2})\big).$$

Thus we have

$$\pi^p_J \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}^m_{k+2})\big) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{34}$$

For, consider the following

$$\begin{aligned} \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) &= \tau_I(x_2) \\ &= \tau_I \circ \pi^p_J \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}^m_{k+2})\big) \\ &= \pi^p_J \circ \tau_I \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}^m_{k+2})\big) && \text{because of (3.12)} \\ &= \pi^p_J \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}^m_{k+2})\big) && \text{see lemma (4.6)} \end{aligned}$$

This means that the first equation of (31) is correct for $l = 0$ for our fixed $k \geq 0$.

Consider the following guarded recursive specification E_3.

$$E_3 = \{\, X_m^3 = t \cdot Y_m^3 + c_4(\overline{k}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i})$$
$$Y_m^3 = c_2(\tilde{d}_{k+2}) \cdot X_{m+1}^3 \;\big|\; m \geq 1 \,\}.$$

Let $\beta_3 = \beta\big(I, \{c_4(\overline{k})\}\big)$. We will show that, if we put for all $m \geq 1$

$$X_m^3 = \pi_J^p \circ \beta_3 \circ \partial_H\big(G_k(\tilde{d}_{k+2}^m)\big)$$

this will solve the guarded recursive specification E_3. We will only verify this for the first equation of E_3

$$\begin{aligned} X_m^3 &= \pi_J^p \circ \beta_3 \circ \partial_H\big(G_k(\tilde{d}_{k+2}^m)\big) \\ &\overset{(27)}{=} t \cdot c_2(\tilde{d}_{k+2}) \cdot \pi_J^p \circ \beta_3 \circ \partial_H\big(G_k(\tilde{d}_{k+2}^{m+1})\big) \\ &\qquad + c_4(\overline{k}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+2}^m)\big) \\ &\overset{(34)}{=} t \cdot Y_m^3 + c_4(\overline{k}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

Again, we will use a collapsing-argument. Consider the guarded recursive specification F_3 without m.

$$F_3 = \{\, X_3 = t \cdot Y_3 + c_4(\overline{k}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i})$$
$$Y_3 = c_2(\tilde{d}_{k+2}) \cdot X_3 \,\}$$

and let (x_3, y_3) be a solution for it. Then it is also a solution for E_3. After using $KFAR_2$ on F_3 we find with (3.12) and (4.6) that

$$\pi_J^p \circ \tau_I \circ \partial_H\big(G_k(\tilde{d}_{k+2}^m)\big) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

So we see that our intermediate result (31) is correct for $l = 0$ since $k \geq 0$ was arbitrarily chosen. Assume that (31) is proved for $l \geq 0$. We will verify it for $l + 1$. Fix a $k \geq 0$ and consider the following guarded recursive specification E_4.

$$E_4 = \{\, X_m^4 = c_2(\tilde{d}_{k+2}) \cdot Y_{m+1}^4 + c_3(\tilde{d}_{k+1}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i})$$
$$Y_{m+1}^4 = t \cdot X_{m+1}^4 + c_3(\tilde{d}_{k+1}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \;\big|\; m \geq 0 \,\}.$$

Let $\beta_4 = \beta\big(I, \{c_3(\tilde{d}_{k+1})\}\big)$. Because of equations (21) and (23) and the induction hypothesis (for l), it is easy to see that if we put for all $m \geq 0$

$$\begin{aligned} X_m^4 &= \pi_J^p \circ \beta_4 \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+1}^{l+1} * \tilde{d}_{k+2}^m)\big) \\ Y_{m+1}^4 &= \pi_J^p \circ \beta_4 \circ \partial_H\big(C_{k+1}(\tilde{d}_{k+1}^{l+1} * \tilde{d}_{k+2}^{m+1})\big), \end{aligned}$$

this will solve the guarded recursive specification E_4. And we see that with the aid of a collapsing-argument we find, in particular,

$$\pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+1}^{l+1} * \tilde{d}_{k+2}^m)\big) = \tau \cdot \prod_{i=1}^p s_5(d_{k+1+i}).$$

Thus, the first equation of (31) is correct for $l+1$. Because of equation (27) and the above display we see that if we put for all $m \geq 1$

$$X_m^3 = \pi_J^p \circ \beta_3 \circ \partial_H\big(G_k(\tilde{d}_{k+1}^{l+1} * \tilde{d}_{k+2}^m)\big)$$

that this will solve the system E_3, too. Hence,

$$\pi_J^p \circ \tau_I \circ \partial_H\big(G_k(\tilde{d}_{k+1}^{l+1} * \tilde{d}_{k+2}^m)\big) = \tau \cdot \prod_{i=1}^p s_5(d_{k+1+i})$$

and our intermediate result (31) is correct for $l+1$. This will end the induction step and we find that (31) is correct.

We were busy with the proof that equation (30) holds if the assumptions on D_k and E_k hold. With the aid of equation (17) we find using our intermediate result for $l = m = 0$

$$\begin{aligned}\pi_J^p \circ \tau_I \circ \partial_H\big(A_{k+1}(\lambda)\big) &= \tau \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\lambda)\big)\\ &= \tau \cdot \prod_{i=1}^p s_5(d_{k+1+i}).\end{aligned}$$

And with equation (18) we find, for all $n \geq 1$, using (31) twice

$$\begin{aligned}\pi_J^p \circ \tau_I \circ \partial_H\big(A_{k+1}(\tilde{d}_{k+1}^n)\big) &= \tau \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+1}^n)\big) + \tau \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(G_k(\tilde{d}_{k+1}^{n-1} * \tilde{d}_{k+2})\big)\\ &= \tau \cdot \prod_{i=1}^p s_5(d_{k+1+i}) + \tau \cdot \prod_{i=1}^p s_5(d_{k+1+i})\\ &= \tau \cdot \prod_{i=1}^p s_5(d_{k+1+i}).\end{aligned}$$

Summarizing, we have seen that for all $n \geq 0$

$$\pi_J^p \circ \tau_I \circ \partial_H\big(A_{k+1}(\tilde{d}_{k+1}^n)\big) = \tau \cdot \prod_{i=1}^p s_5(d_{k+1+i}). \tag{35}$$

Now consider the following guarded recursive specification E_5.

$$\begin{aligned}E_5 = \Big\{\, X_n^5 &= t \cdot Y_n^5 + c_4(\overline{k}) \cdot \prod_{i=1}^p s_5(d_{k+1+i})\\ Y_n^5 &= c_2(\tilde{d}_{k+1}) \cdot X_{n+1}^5 \;\Big|\; n \geq 0 \,\Big\}.\end{aligned}$$

With the aid of equation (26) and (35), it is evident that if we put for all $n \geq 0$

$$X_n^5 = \pi_J^p \circ \beta_3 \circ \partial_H\big(F_k(\tilde{d}_{k+1}^n)\big)$$

this is a solution for the system E_5. Using a collapsing-argument we will find immediately that

$$\pi_J^p \circ \tau_I \circ \partial_H\big(F_k(\tilde{d}_{k+1}^n)\big) = \tau \cdot \prod_{i=1}^p s_5(d_{k+1+i}).$$

This is precisely equation (30). Thus, we have proved that if equations (28) and (29) hold that (30) also holds. Now we are going to prove that equation (30) itself holds. We will do that with induction on p. So let $p = 1$. Observe that equations (28) and (29) are valid for $p = 1$ (q.v. equations (24) and (25)). Thus, we find that equation (30) is valid for $p = 1$. Now suppose that (30) holds for $p \geq 1$. Then it is easy to see with the aid of equation (24) and the induction hypothesis for p that

$$\begin{aligned}\pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^n)\big) &= t \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) + s_5(d_{k+1}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ &= t \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) + \prod_{i=1}^{p+1} s_5(d_{k+i}).\end{aligned}$$

This is equation (28) for $p+1$. It can be seen in exactly the same way that equation (29) is also valid for $p+1$. But that means that equation (30) is correct for $p+1$. So this will end the induction step and we find that (30) is true for all $p \geq 1$*. Consider the following guarded recursive specification E_6.

$$\begin{aligned}E_6 = \Big\{\, X_n^6 &= t \cdot Y_n^6 + \prod_{i=1}^{p} s_5(d_{k+i}) \\ Y_n^6 &= c_2(\tilde{d}_{k+1}) \cdot X_{n+1}^6 + \prod_{i=1}^{p} s_5(d_{k+i}) \;\Big|\; n \geq 0 \,\Big\}.\end{aligned}$$

With the aid of equations (24) and (25) we can conclude, using equation (30) that, if we put for all $n \geq 0$

$$\begin{aligned}X_n^6 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^n)\big) \\ Y_n^6 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big)\end{aligned}$$

this is a solution for E_6. With a collapsing-argument we find that

$$\left.\begin{aligned}\pi_J^p \circ \tau_I \circ \partial_H\big(D_k(\tilde{d}_{k+1}^n)\big) \\ \pi_J^p \circ \tau_I \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big)\end{aligned}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+i}). \tag{36}$$

Consider the guarded recursive specification E_7.

$$\begin{aligned}E_7 = \Big\{\, X_n^7 &= c_2(\tilde{d}_{k+1}) \cdot Y_{n+1}^7 + c_3(\tilde{d}_{k+1}) \cdot \prod_{i=1}^{p} s_5(d_{k+i}) \\ Y_{n+1}^7 &= t \cdot X_{n+1}^7 + c_3(\tilde{d}_{k+1}) \cdot \prod_{i=1}^{p} s_5(d_{k+i}) \;\Big|\; n \geq 1 \,\Big\}.\end{aligned}$$

It is easy to see with the aid of equations (20), (22) and (36) that, if we put for all $n \geq 1$

$$\begin{aligned}X_n^7 &= \pi_J^p \circ \beta_4 \circ \partial_H\big(B_k(\tilde{d}_{k+1}^n)\big) \\ Y_{n+1}^7 &= \pi_J^p \circ \beta_4 \circ \partial_H\big(C_k(\tilde{d}_{k+1}^{n+1})\big),\end{aligned}$$

this will solve the system E_7. So, in particular, we find, using a collapsing-argument, for all $n \geq 1$

$$\pi_J^p \circ \tau_I \circ \partial_H\big(C_k(\tilde{d}_{k+1}^n)\big) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+i}). \tag{37}$$

* Notice that now equations (28) and (29) are true for all $p \geq 1$, as well.

Now we can find with the aid of equation (17), (19) and (37) that for $k = 0$

$$\begin{aligned}\pi_J^p \circ \tau_I \circ \partial_H\big(A_0(\lambda)\big) &= \tau \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(B_0(\lambda)\big) \\ &= \tau \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(C_0(\tilde{d}_1)\big) \\ &= \tau \cdot \prod_{i=1}^{p} s_5(d_i).\end{aligned}$$

Thus, we have proved for all $p \geq 1$

$$\pi_J^p \circ \tau_I \circ \partial_H\big(A_0(\lambda)\big) = \tau \cdot \pi_J^p\Big(\prod_{i=1}^{\infty} s_5(d_i)\Big)$$

and with a *GIP* argument we can conclude the proof.

Remark (4.9)

In this verification, we have seen the frequent use of a so-called collapsing-argument. We can imagine that if the protocol becomes more complicated the collapsing-argument will become more complicated, as well. We can provide for each situation a new collapsing-argument; but it will be better to formulate a general theorem on collapsing-arguments that we can use in every situation. Fortunately, such a theorem already exists. Hereinafter, we will enumerate all the definitions needed in order to state this theorem.

The following definitions are taken from [**17**].

Definition (4.10)

Let $E = \{x = t_x(X) : x \in X\}$ be a guarded recursive specification (without abstracting operators). Let $I \subseteq A$. We will call a subset $C \subseteq X$ a cluster of I in E if $C \neq \emptyset$ and for all $x \in C$ we have $i_1, \ldots, i_m \in I \cup \{\tau\}$, $x_1, \ldots, x_m \in C$ and $y_1, \ldots, y_n \in X \setminus C$ $(n \geq 1, m \geq 0)$ such that the equation for x in the guarded recursive specification E is

$$x = \sum_{k=1}^{m} i_k \cdot x_k + \sum_{l=1}^{n} y_l,$$

in which the last summand is omitted if $n = 0$. We will call the variables y_l exits of x and we write $U(x) = \{y_1, \ldots, y_n\}$. We will use $U(C) = \bigcup_{x \in X} U(x)$ for the exit set of the cluster C. We will call a cluster *conservative* if every exit $y \in U(C)$ is accessible from every variable in the cluster by doing zero or more steps from $I \cup \{\tau\}$ in the cluster to a cluster-variable which has exit y.

Definition (4.11) *Cluster Fair Abstraction Rule*

Let E be a guarded recursive specification and let $I \subseteq A$ with $|I| \geq 2$. Let C be a conservative cluster of E in I. Let $|U(C)| < \infty$. Then we have for all $x \in C$:

$$\tau_I(x) = \tau \cdot \sum_{y \in U(C)} \tau_I(y).$$

We will refer to this rule with the abbreviation *CFAR*$^\infty$, since it is not supposed that the cluster itself is finite; only the exit set of the cluster. If C is a finite set then the abbreviation *CFAR* will be used.

The following theorem is taken from [**17**]. Only some notations are modified.

Theorem (4.12)

$$ACP_{\tau,u} + RDP + RSP + ODP + OSP + KFAR_1 + GIP \vdash CFAR^{\infty}.$$

Example (4.13)

Let us take a look at the guarded recursive specification E_1 that can be found in the proof of theorem (4.8).

$$\begin{aligned} E_1 = \{\, X_n^1 &= t \cdot Y_n^1 + Z \\ Y_n^1 &= c_2(\tilde{d}_{k+2}) \cdot X_{n+1}^1 + Z \\ Z &= \prod_{i=1}^{p} s_5(d_{k+1+i}) \bigm| n \geq 0 \,\}. \end{aligned}$$

It is not very hard to see that $C = \{X_n^1, Y_n^1 : n \geq 0\}$ is a conservative cluster of E_1 in I and that the set of exits $U(C) = \{Z\}$. Hence, we may apply $CFAR^{\infty}$ and we find thus for all $n \geq 0$

$$\tau_I(X_n^1) = \tau_I(Y_n^1) = \tau \cdot \tau_I(Z) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

So, in fact, our collapsing-argument is just an application of $CFAR^{\infty}$.

In our next example we will make things a little less unrealistic. We will change the behaviour of the queue Q that already appeared in the specification of our alternating bit protocol: the queue can lose its last frame. We will express this with an atomic action l (for *lost*) as follows.

$$\begin{aligned} Q(\lambda) &= \sum_{x \in E} r_2(x) \cdot Q(x) \\ Q(x * \sigma) &= \sum_{y \in E} r_2(y) \cdot Q(x * \sigma * y) + s_3(x) \cdot Q(\sigma) + l \cdot Q(\sigma). \end{aligned}$$

All the other components of our alternating bit protocol will remain the same and will be used in the following theorem. Observe that we use the same symbol for this faulty queue. This is done because abbreviations in the former theorem can be used again. We will enumerate the following sets that will be used in this theorem.

$$\begin{aligned} H = &\{\, r_1(d), s_1(d) : d \in D \} \\ &\cup \{\, r_2(x), r_3(x), s_2(x), s_3(x) : x \in E \} \\ &\cup \{\, r_4(\overline{k}), s_4(\overline{k}) : \overline{k} \in \mathbb{Z}/2\mathbb{Z} \,\} \\ I = &\{\, c_1(d) : d \in D \,\} \\ &\cup \{\, c_2(x), c_3(x) : x \in E \,\} \\ &\cup \{\, c_4(\overline{k}) : \overline{k} \in \mathbb{Z}/2\mathbb{Z} \,\} \cup \{l, t\}. \end{aligned}$$

Notice that the only difference with the sets used in theorem (4.8) is the extra l in the abstraction set I.

Theorem (4.14)

Let Q be the above queue with the possibility that the last frame can be lost. The other components are the same that appear in theorem (4.8).

$$\tau_I \circ \partial_H\Big(S(\overline{0}) \parallel Q(\lambda) \parallel R(\overline{0}) \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big) = \tau \cdot \prod_{i=1}^{\infty} s_5(d_i).$$

Proof. The structure of this proof will be the same as before. We will first calculate

$$\partial_H\Big(S(\overline{0}) \parallel Q(\lambda) \parallel R(\overline{0}) \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big).$$

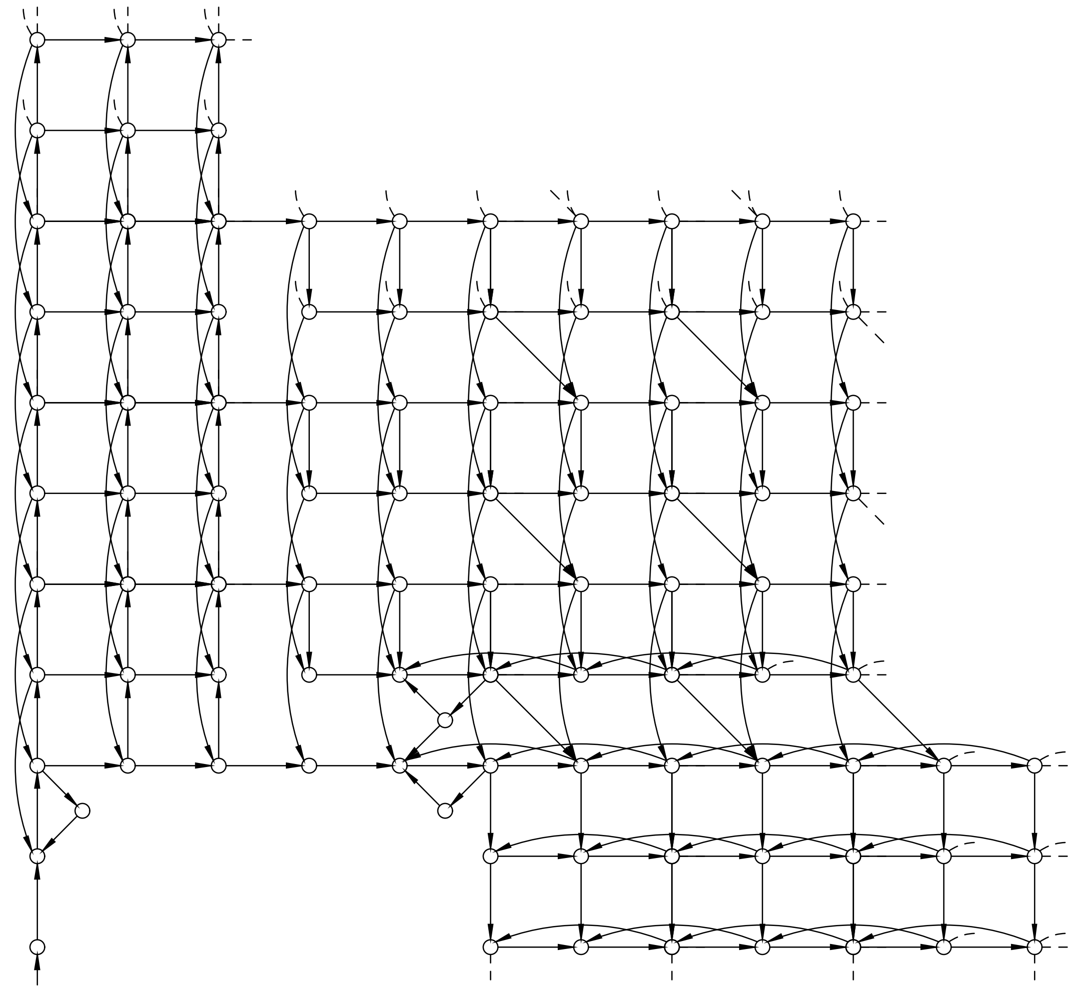

Figure 9. A state transition diagram.

We have included a picture of this process in figure 9 on page 40. The actual calculation of this process will be left to the reader, but we will provide some explanation such that with the aid of the picture and the specification it will be not that hard to verify that the result is correct. We will use the same notations as in theorem (4.8). In order to give the specification we will need three more abbreviations.

$$H_k(\sigma) = S(\tilde{d}_{k+1}) \parallel Q(\sigma) \parallel SA(\overline{k}) \parallel \prod_{i=k+2}^{\infty} s_1(d_i)$$

$$I_k(\sigma) = S(\overline{k+1}) \parallel Q(\sigma) \parallel SA(\overline{k}) \parallel \prod_{i=k+2}^{\infty} s_1(d_i)$$

$$J_k(\sigma) = S(\tilde{d}_{k+2}) \parallel Q(\sigma) \parallel SA(\overline{k}) \parallel \prod_{i=k+3}^{\infty} s_1(d_i)$$

Hereinafter, we will enumerate the guarded recursive specification. We have $k \geq 0$ and $n \geq 1$. For m we will be explicit. Then we will discuss this result.

$$\partial_H\big(A_k(\lambda)\big) = c_1(d_{k+1}) \cdot \partial_H\big(B_k(\lambda)\big) \tag{38}$$

$$
\begin{aligned}
\partial_H\big(A_{k+1}(\tilde{d}^{n}_{k+1})\big) &= l\cdot\partial_H\big(A_{k+1}(\tilde{d}^{n-1}_{k+1})\big) + c_1(d_{k+2})\cdot\partial_H\big(B_{k+1}(\tilde{d}^{n}_{k+1})\big)\\
&\quad + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(I_k(\tilde{d}^{n-1}_{k+1})\big) && (39)
\end{aligned}
$$

$$
\begin{aligned}
\partial_H\big(B_k(\lambda)\big) &= c_2(\tilde{d}_{k+1})\cdot\partial_H\big(C_k(\tilde{d}_{k+1})\big) && (40)\\
\partial_H\big(B_k(\tilde{d}^{n}_{k+1})\big) &= l\cdot\partial_H\big(B_k(\tilde{d}^{n-1}_{k+1})\big) + c_2(\tilde{d}_{k+1})\cdot\partial_H\big(C_k(\tilde{d}^{n+1}_{k+1})\big)\\
&\quad + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(E_k(\tilde{d}^{n-1}_{k+1})\big) && (41)\\
\partial_H\big(B_{k+1}(\tilde{d}^{n}_{k+1} * \tilde{d}^{m}_{k+2})\big) &= l\cdot\partial_H\big(B_{k+1}(\tilde{d}^{n-1}_{k+1} * \tilde{d}^{m}_{k+2})\big) && (m\ge 0)\\
&\quad + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(J_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^{m}_{k+2})\big)\\
&\quad + c_2(\tilde{d}_{k+2})\cdot\partial_H\big(C_{k+1}(\tilde{d}^{n}_{k+1} * \tilde{d}^{m+1}_{k+2})\big) && (42)
\end{aligned}
$$

$$
\begin{aligned}
\partial_H\big(C_k(\lambda)\big) &= t\cdot\partial_H\big(B_k(\lambda)\big) && (43)\\
\partial_H\big(C_k(\tilde{d}^{n}_{k+1})\big) &= l\cdot\partial_H\big(C_k(\tilde{d}^{n-1}_{k+1})\big) + t\cdot\partial_H\big(B_k(\tilde{d}^{n}_{k+1})\big)\\
&\quad + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(D_k(\tilde{d}^{n-1}_{k+1})\big) && (44)\\
\partial_H\big(C_{k+1}(\tilde{d}^{n}_{k+1} * \tilde{d}^{m}_{k+2})\big) &= l\cdot\partial_H\big(C_{k+1}(\tilde{d}^{n-1}_{k+1} * \tilde{d}^{m}_{k+2})\big) && (m\ge 1)\\
&\quad + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(G_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^{m}_{k+2})\big)\\
&\quad + t\cdot\partial_H\big(B_{k+1}(\tilde{d}^{n}_{k+1} * \tilde{d}^{m}_{k+2})\big) && (45)
\end{aligned}
$$

$$
\begin{aligned}
\partial_H\big(D_k(\lambda)\big) &= t\cdot\partial_H\big(E_k(\lambda)\big) + s_5(d_{k+1})\cdot\partial_H\big(F_k(\lambda)\big) && (46)\\
\partial_H\big(D_k(\tilde{d}^{n}_{k+1})\big) &= l\cdot\partial_H\big(D_k(\tilde{d}^{n-1}_{k+1})\big) + t\cdot\partial_H\big(E_k(\tilde{d}^{n}_{k+1})\big)\\
&\quad + s_5(d_{k+1})\cdot\partial_H\big(F_k(\tilde{d}^{n}_{k+1})\big) && (47)
\end{aligned}
$$

$$
\begin{aligned}
\partial_H\big(E_k(\lambda)\big) &= c_2(\tilde{d}_{k+1})\cdot\partial_H\big(D_k(\tilde{d}_{k+1})\big) + s_5(d_{k+1})\cdot\partial_H\big(H_k(\lambda)\big) && (48)\\
\partial_H\big(E_k(\tilde{d}^{n}_{k+1})\big) &= l\cdot\partial_H\big(E_k(\tilde{d}^{n-1}_{k+1})\big) + c_2(\tilde{d}_{k+1})\cdot\partial_H\big(D_k(\tilde{d}^{n+1}_{k+1})\big)\\
&\quad + s_5(d_{k+1})\cdot\partial_H\big(H_k(\tilde{d}^{n}_{k+1})\big) && (49)
\end{aligned}
$$

$$
\begin{aligned}
\partial_H\big(F_k(\lambda)\big) &= t\cdot\partial_H\big(H_k(\lambda)\big) + c_4(\overline{k})\cdot\partial_H\big(A_{k+1}(\lambda)\big) && (50)\\
\partial_H\big(F_k(\tilde{d}^{n}_{k+1})\big) &= l\cdot\partial_H\big(F_k(\tilde{d}^{n-1}_{k+1})\big) + t\cdot\partial_H\big(H_k(\tilde{d}^{n}_{k+1})\big)\\
&\quad + c_4(\overline{k})\cdot\partial_H\big(A_{k+1}(\tilde{d}^{n}_{k+1})\big) && (51)
\end{aligned}
$$

$$
\begin{aligned}
\partial_H\big(G_k(\lambda)\big) &= t\cdot\partial_H\big(J_k(\lambda)\big) + c_4(\overline{k})\cdot\partial_H\big(B_{k+1}(\lambda)\big) && (52)\\
\partial_H\big(G_k(\tilde{d}^{m}_{k+2})\big) &= l\cdot\partial_H\big(G_k(\tilde{d}^{m-1}_{k+2})\big) + t\cdot\partial_H\big(J_k(\tilde{d}^{m}_{k+2})\big) && (m\ge 1)\\
&\quad + c_4(\overline{k})\cdot\partial_H\big(B_{k+1}(\tilde{d}^{m}_{k+2})\big) && (53)\\
\partial_H\big(G_k(\tilde{d}^{n}_{k+1} * \tilde{d}^{m}_{k+2})\big) &= l\cdot\partial_H\big(G_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^{m}_{k+2})\big) + t\cdot\partial_H\big(J_k(\tilde{d}^{n}_{k+1} * \tilde{d}^{m}_{k+2})\big) && (m\ge 1)\\
&\quad + c_4(\overline{k})\cdot\partial_H\big(B_{k+1}(\tilde{d}^{n}_{k+1} * \tilde{d}^{m}_{k+2})\big) && (54)
\end{aligned}
$$

$$
\begin{aligned}
\partial_H\big(H_k(\lambda)\big) &= c_2(\tilde{d}_{k+1})\cdot\partial_H\big(F_k(\tilde{d}_{k+1})\big) && (55)\\
\partial_H\big(H_k(\tilde{d}^{n}_{k+1})\big) &= l\cdot\partial_H\big(H_k(\tilde{d}^{n-1}_{k+1})\big) + c_2(\tilde{d}_{k+1})\cdot\partial_H\big(F_k(\tilde{d}^{n+1}_{k+1})\big) && (56)
\end{aligned}
$$

$$
\begin{aligned}
\partial_H\big(I_k(\lambda)\big) &= c_1(d_{k+2})\cdot\partial_H\big(J_k(\lambda)\big) && (57)\\
\partial_H\big(I_k(\tilde{d}^{n}_{k+1})\big) &= l\cdot\partial_H\big(I_k(\tilde{d}^{n-1}_{k+1})\big) + c_1(d_{k+2})\cdot\partial_H\big(J_k(\tilde{d}^{n}_{k+1})\big) && (58)
\end{aligned}
$$

$$
\begin{aligned}
\partial_H\big(J_k(\lambda)\big) &= c_2(\tilde{d}_{k+2})\cdot\partial_H\big(G_k(\tilde{d}_{k+2})\big) && (59)\\
\partial_H\big(J_k(\tilde{d}^{m}_{k+2})\big) &= l\cdot\partial_H\big(J_k(\tilde{d}^{m-1}_{k+2})\big) && (m\ge 1)\\
&\quad + c_2(\tilde{d}_{k+2})\cdot\partial_H\big(G_{k+1}(\tilde{d}^{m+1}_{k+2})\big) && (60)\\
\partial_H\big(J_k(\tilde{d}^{n}_{k+1} * \tilde{d}^{m}_{k+2})\big) &= l\cdot\partial_H\big(J_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^{m}_{k+2})\big) && (m\ge 0)\\
&\quad + c_2(\tilde{d}_{k+2})\cdot\partial_H\big(G_{k+1}(\tilde{d}^{n}_{k+1} * \tilde{d}^{m+1}_{k+2})\big) && (61)
\end{aligned}
$$

At this point, we will clarify the guarded recursive specification consisting of the equations (38)–(61). We are to give a guarded recursive specification of the process $\partial_H\big(A_0(\lambda)\big)$. If we calculate this, we can see in figure 9, starting from the root of the graph, that we will rise to the next node, which is $\partial_H\big(B_0(\lambda)\big)$. This part of the picture corresponds to equation (38) for $k = 0$. If we calculate this node we will go up again and we will enter "the single* *BC*-trace", which corresponds to equations (41) and (44). We can enter a triangle from this node that corresponds to equations (44) for $n = 1$ and equation (43). If we go one step to the left from the single *BC*-trace we will enter the *DE*-trace, which corresponds to the equations (46)–(49). In the *DE*-trace we can do an $s_5(d_0)$ step to the left and then we will enter the *FH*-trace. This trace resembles the equations (50), (51), (55) and (56). If we step to the left in the *FH*-trace on the lowest level, (see equation (50)), we see that we will enter $\partial_H\big(A_1(\lambda)\big)$, via an acknowledgement $c_4(\overline{0})$. This corresponds to equation (38) for $k = 1$. Thereupon, we will enter the single *BC*-trace for $k = 1$, or a triangle, and so on. If we do a step to the left from the *FH*-trace on another level, we will enter an A-node, which corresponds to equation (39) (confer (51)). At this A-node we have three choices: we can jump to lower A-node, we can go down to an I-node (57) and (58), or we can enter a double** *BC*-trace, which corresponds to the equations (42) and (45). From the double *BC*-trace we can do a step down and enter the *JG*-trace or jump down to a lower *BC*-trace. This can be a double *BC*-trace, or if we are on the first level, a single one. From the I-node we can jump to a lower I-node, except on the first level, or we can enter the *JG*-trace, as well. This *JG*-trace corresponds to equations (52)–(54) and (59)–(61). From the *JG*-trace we can jump down to a lower *JG*-trace, except one the first level; there we can only jump to the right in the same *JG*-trace. Or we can do a "skewed" step down to a lower *BC*-trace, which can be both double or single. There is one node that we have not met yet: it is the node between the lowest *JG*-trace and the single *BC*-trace right under it. This node corresponds to equation (52).

Now we have seen how this big guarded recursive specification can be deduced, we will briefly discuss the overall structure of the sequel: first, we will make an assumption for one $p \geq 1$ on a *DE*-trace, from which we already know that it holds for $p = 1$. With this assumption we can derive a conclusion for an *FH*-trace. To achieve this we will need an intermediary result that we will prove with induction. Once we derived the conclusion on this *FH*-trace we will set off a chain reaction: we know that the assumption on the *DE*-trace holds for $p = 1$ and the we conclude that the conclusion on the *FH*-trace is valid for $p = 1$. With this knowledge we can derive that the assumption on the *DE*-trace is correct for $p = 2$ and so we find that the conclusion concerning the *FH*-trace holds for $p = 2$ et cetera. This will be done with induction on p. In the end it will turn out that both the assumption and the conclusion hold for all $p \geq 1$. This fact will be used to make a small calculation on a single *BC*-trace. Using this together with a *GIP* argument, we can conclude the proof of (4.14).

Let $J = \{s_5(d) : d \in D\}$. Let $\beta_1 = \beta(I, J)$. Let, for the moment, $p \geq 1$ be fixed. Suppose that the following holds. For all $k \geq 0$ and for all $n \geq 1$:

$$\pi_J^p \circ \beta_1 \circ \partial_H\big(D_k(\lambda)\big) = t \cdot \pi_J^p \circ \beta_1 \circ \partial_H\big(E_k(\lambda)\big) + \prod_{i=1}^{p} s_5(d_{k+i}) \tag{62}$$

$$\begin{aligned}\pi_J^p \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^n)\big) = {} & t \cdot \pi_J^p \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) + l \cdot \pi_J^p \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^{n-1})\big) \\ & + \prod_{i=1}^{p} s_5(d_{k+i})\end{aligned} \tag{63}$$

$$\pi_J^p \circ \beta_1 \circ \partial_H\big(E_k(\lambda)\big) = c_2(\tilde{d}_{k+1}) \cdot \pi_J^p \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1})\big) + \prod_{i=1}^{p} s_5(d_{k+i}) \tag{64}$$

$$\begin{aligned}\pi_J^p \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) = {} & c_2(\tilde{d}_{k+1}) \cdot \pi_J^p \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^{n+1})\big) + l \cdot \pi_J^p \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^{n-1})\big) \\ & + \prod_{i=1}^{p} s_5(d_{k+i}).\end{aligned} \tag{65}$$

* Single stands for the fact that there is only one type of elements in the queue.

** Double stands for the fact that there are two types of elements in the queue.

Then the following can be derived from this assumption. For all $k, n \geq 0$:

$$\left.\begin{aligned}\pi_J^p \circ \tau_I \circ \partial_H\big(F_k(\tilde{d}_{k+1}^n)\big)\\ \pi_J^p \circ \tau_I \circ \partial_H\big(H_k(\tilde{d}_{k+1}^n)\big)\end{aligned}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{66}$$

To prove this we will first deduce from the assumptions that we made an intermediary result on the JG-traces and the double BC-traces. Hereinafter, we will state it and we will prove it with induction on u. For all $k, u, m \geq 0$:

$$\left.\begin{aligned}\pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+1}^u * \tilde{d}_{k+2}^m)\big)\\ \pi_J^p \circ \tau_I \circ \partial_H\big(C_{k+1}(\tilde{d}_{k+1}^u * \tilde{d}_{k+2}^{m+1})\big)\\ \pi_J^p \circ \tau_I \circ \partial_H\big(G_k(\tilde{d}_{k+1}^u * \tilde{d}_{k+2}^{m+1})\big)\\ \pi_J^p \circ \tau_I \circ \partial_H\big(J_k(\tilde{d}_{k+1}^u * \tilde{d}_{k+2}^m)\big)\end{aligned}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \tag{67}$$

As we already announced, we will prove (67) with induction on u. So let first of all $u = 0$ and fix $k \geq 0$. Consider the following guarded recursive specification E_1.

$$\begin{aligned} E_1 = \big\{\, X_0^1 &= t \cdot Y_0^1 + Z_1 \\ Y_0^1 &= c_2(\tilde{d}_{k+2}) \cdot X_1^1 + Z_1 \\ X_n^1 &= l \cdot X_{n-1}^1 + t \cdot Y_n^1 + Z_1 \\ Y_n^1 &= l \cdot Y_{n-1}^1 + c_2(\tilde{d}_{k+2}) \cdot X_{n+1}^1 + Z_1 \\ Z_1 &= \prod_{i=1}^{p} s_5(d_{k+1+i}) \;\big|\; n \geq 1 \,\big\}. \end{aligned}$$

Let $C_1 = \{X_n^1, Y_n^1 : n \geq 0\}$. Then C_1 is a conservative cluster of E_1 in I. Since the set of exits $U(C_1) = \{Z_1\}$ is finite, we may apply $CFAR^\infty$ and we find for all $n \geq 0$:

$$\tau_I(X_n^1) = \tau_I(Y_n^1) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Because of equations (62)–(65) we see that if we put for all $n \geq 0$

$$\begin{aligned} X_n^1 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) \\ Y_n^1 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(E_{k+1}(\tilde{d}_{k+2}^n)\big), \end{aligned}$$

this will solve the guarded recursive specification E_1. Now consider the following calculation.

$$\begin{aligned} \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) &= \tau_I(X_n^1) \\ &= \tau_I \circ \pi_J^p \circ \beta_1 \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) \\ &= \pi_J^p \circ \tau_I \circ \beta_1 \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) && \text{because of (3.12)} \\ &= \pi_J^p \circ \tau_I \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big) && \text{see lemma (4.6)} \end{aligned}$$

Such a calculation can be made for E_{k+1}, too. We find thus for all $n \geq 0$

$$\left.\begin{aligned}\pi_J^p \circ \tau_I \circ \partial_H\big(D_{k+1}(\tilde{d}_{k+2}^n)\big)\\ \pi_J^p \circ \tau_I \circ \partial_H\big(E_{k+1}(\tilde{d}_{k+2}^n)\big)\end{aligned}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{68}$$

Now consider the following guarded recursive specification E_2.

$$\begin{aligned} E_2 = \{\, X_0^2 &= c_2(\tilde{d}_{k+2}) \cdot Y_1^2 \\ Y_0^2 &= t \cdot X_0^2 \\ X_n^2 &= l \cdot X_{n-1}^2 + c_2(\tilde{d}_{k+2}) \cdot Y_{n+1}^2 + Z_2 \\ Y_n^2 &= l \cdot Y_{n-1}^2 + t \cdot X_n^2 + Z_2 \\ Z_2 &= c_3(\tilde{d}_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \;\Big|\; n \geq 1 \,\}. \end{aligned}$$

Let $C_2 = \{X_n^2, Y_n^2 : n \geq 0\}$. Then C_2 is a conservative cluster of E_2 in I. Since the set of exits $U(C_2) = \{Z_2\}$ is finite, we may apply $CFAR^\infty$ and we find for all $n \geq 0$:

$$\tau_I(X_n^2) = \tau_I(Y_n^2) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Let $\beta_2 = \beta\big(I, \{c_3(\tilde{d}_{k+2})\}\big)$. Hereinafter, we will show that if we put for all $n \geq 0$

$$\begin{aligned} X_n^2 &= \pi_J^p \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+2}^n)\big) \\ Y_n^2 &= \pi_J^p \circ \beta_2 \circ \partial_H\big(C_{k+1}(\tilde{d}_{k+2}^n)\big), \end{aligned} \tag{69}$$

this will solve the guarded recursive specification E_2. For $n = 0$ it is trivial to see that, with the aid of equations (40) and (43), this is true for the first two equations of E_2. Now let $n \geq 1$. We will verify the third equation of E_2.

$$\begin{aligned} \pi_J^p \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+2}^n)\big) &\overset{(41)}{=} l \cdot \pi_J^p \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+2}^{n-1})\big) \\ &\quad + c_2(\tilde{d}_{k+2}) \cdot \pi_J^p \circ \beta_2 \circ \partial_H\big(C_{k+1}(\tilde{d}_{k+2}^{n+1})\big) \\ &\quad + c_3(\tilde{d}_{k+2}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(E_{k+1}(\tilde{d}_{k+2}^{n-1})\big) \\ &\overset{(68)}{=} l \cdot \pi_J^p \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+2}^{n-1})\big) \\ &\quad + c_2(\tilde{d}_{k+2}) \cdot \pi_J^p \circ \beta_2 \circ \partial_H\big(C_{k+1}(\tilde{d}_{k+2}^{n+1})\big) \\ &\quad + c_3(\tilde{d}_{k+2}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

So we see that this solves the equation in E_2 for X_n^2. A similar calculation will yield the same result for the fourth equation of E_2. So the claim that (69) is a solution for E_2 is valid. We find

$$\begin{aligned} \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) &= \tau_I(X_n^2) \\ &= \tau_I \circ \pi_J^p \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+2}^n)\big) \\ &= \pi_J^p \circ \tau_I \circ \beta_2 \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+2}^n)\big) && \text{because of (3.12)} \\ &= \pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+2}^n)\big). && \text{see lemma (4.6)} \end{aligned}$$

Such a calculation can be made for C_{k+1}, too. Summarizing we have for all $n \geq 0$

$$\left.\begin{aligned} \pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+2}^n)\big) \\ \pi_J^p \circ \tau_I \circ \partial_H\big(C_{k+1}(\tilde{d}_{k+2}^n)\big) \end{aligned}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{70}$$

This means that the first two equations of (67) are proved for $u = 0$ and our fixed $k \geq 0$. Consider the next guarded recursive specification E_3.

$$\begin{aligned} E_3 = \big\{\, X_0^3 &= t \cdot Y_0^3 + Z_3 \\ Y_0^3 &= c_2(\tilde{d}_{k+2}) \cdot X_1^3 \\ X_m^3 &= l \cdot X_{m-1}^3 + t \cdot Y_m^3 + Z_3 \\ Y_m^3 &= l \cdot Y_{m-1}^3 + c_2(\tilde{d}_{k+2}) \cdot X_{m+1}^3 \\ Z_3 &= c_4(\overline{k}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \;\big|\; n \geq 1 \,\big\}. \end{aligned}$$

Let $C_3 = \{X_m^3, Y_m^3 : m \geq 0\}$. Then C_3 is a conservative cluster of E_3 in I. Since the set of exits $U(C_3) = \{Z_3\}$ is finite, we may apply $CFAR^\infty$ and we find for all $n \geq 0$:

$$\tau_I(X_m^3) = \tau_I(Y_m^3) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Let $\beta_3 = \beta\big(I, \{c_4(\overline{k})\}\big)$. It will not be difficult to see that if we put for all $m \geq 0$

$$\begin{aligned} X_m^3 &= \pi_J^p \circ \beta_3 \circ \partial_H\big(G_k(\tilde{d}_{k+2}^m)\big) \\ Y_m^3 &= \pi_J^p \circ \beta_3 \circ \partial_H\big(J_k(\tilde{d}_{k+2}^m)\big), \end{aligned} \tag{71}$$

this will solve the system E_3. We will only verify the first equation for E_3. Consider thereunto the following display.

$$\begin{aligned} \pi_J^p \circ \beta_3 \circ \partial_H\big(G_k(\lambda)\big) &\overset{(52)}{=} t \cdot \pi_J^p \circ \beta_3 \circ \partial_H\big(J_k(\lambda)\big) + c_4(\overline{k}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\lambda)\big) \\ &\overset{(70)}{=} t \cdot \pi_J^p \circ \beta_3 \circ \partial_H\big(J_k(\lambda)\big) + c_4(\overline{k}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

The other equations can be handled in the same way while using equation (70). Thus, we find that (71) solves the guarded recursive specification E_3. When we use the fact that the generalized projection operator and the abstraction operator commute and the fact that the abstraction operator absorbs the selective abstraction operator β_3 we will find with a calculation that is similar to the ones we did for E_1 and E_2 that for all $m \geq 0$

$$\left.\begin{aligned} &\pi_J^p \circ \tau_I \circ \partial_H\big(G_k(\tilde{d}_{k+2}^m)\big) \\ &\pi_J^p \circ \tau_I \circ \partial_H\big(J_k(\tilde{d}_{k+2}^m)\big) \end{aligned}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Now we find that the second two equations of (67) are also correct so we see that the basis of the induction is proved for (67), since $k \geq 0$ was arbitrarily chosen. Assume that (67) is valid for $u \geq 0$. We will prove it for $u + 1$. Fix $k \geq 0$ and let E_4 be the following guarded recursive specification.

$$\begin{aligned} E_4 = \big\{\, X_m^4 &= c_2(\tilde{d}_{k+2}) \cdot Y_{m+1}^4 + Z_4 \\ Y_{m+1}^4 &= t \cdot X_{m+1}^4 + Z_4 \\ Z_4 &= \big(c_3(\tilde{d}_{k+1}) + l\big) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \;\big|\; m \geq 0 \,\big\}. \end{aligned}$$

Let $C_4 = \{X_m^4, Y_{m+1}^4 : m \geq 0\}$. Then it is easy to see that C_4 is a conservative cluster of E_4 in I. The set of exits $U(C_4) = \{Z_4\}$ is finite, so we may apply $CFAR^\infty$ and we find for all $m \geq 0$

$$\tau_I(X_m^4) = \tau_I(Y_{m+1}^4) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Let $\beta_4 = \beta\big(I, \{c_3(\tilde{d}_{k+1}), l\}\big)$. We will show that if we put for all $m \geq 0$

$$\begin{aligned} X^4_m &= \pi^p_J \circ \beta_4 \circ \partial_H\big(B_{k+1}(\tilde{d}^{u+1}_{k+1} * \tilde{d}^m_{k+2})\big) \\ Y^4_{m+1} &= \pi^p_J \circ \beta_4 \circ \partial_H\big(C_{k+1}(\tilde{d}^{u+1}_{k+1} * \tilde{d}^{m+1}_{k+2})\big), \end{aligned} \tag{72}$$

this will be a solution for the system E_4. For, contemplate the deduction below concerning the first equation of E_4.

$$\begin{aligned} \pi^p_J \circ \beta_4 \circ \partial_H\big(B_{k+1}(\tilde{d}^{u+1}_{k+1} * \tilde{d}^m_{k+2})\big) &\overset{(42)}{=} l \cdot \pi^p_J \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}^{u}_{k+1} * \tilde{d}^m_{k+2})\big) \\ &\quad + c_2(\tilde{d}_{k+2}) \cdot \pi^p_J \circ \beta_4 \circ \partial_H\big(C_{k+1}(\tilde{d}^{u+1}_{k+1} * \tilde{d}^{m+1}_{k+2})\big) \\ &\quad + c_3(\tilde{d}_{k+1}) \cdot \pi^p_J \circ \tau_I \circ \partial_H\big(J_k(\tilde{d}^{u}_{k+1} * \tilde{d}^m_{k+2})\big) \\ &\overset{(67)}{=} l \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) + c_3(\tilde{d}_{k+1}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ &\quad + c_2(\tilde{d}_{k+2}) \cdot \pi^p_J \circ \beta_4 \circ \partial_H\big(C_{k+1}(\tilde{d}^{u+1}_{k+1} * \tilde{d}^{m+1}_{k+2})\big) \\ &= c_2(\tilde{d}_{k+2}) \cdot \pi^p_J \circ \beta_4 \circ \partial_H\big(C_{k+1}(\tilde{d}^{u+1}_{k+1} * \tilde{d}^{m+1}_{k+2})\big) \\ &\quad + \big(c_3(\tilde{d}_{k+1}) + l\big) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

Completely analogously to the above calculation we can deduce a similar result for the second equation of E_4. So we see that the solution for the guarded recursive specification E_4 is (72). We find easily using commutativity and absorption:

$$\left.\begin{aligned} &\pi^p_J \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}^{u+1}_{k+1} * \tilde{d}^m_{k+2})\big) \\ &\pi^p_J \circ \tau_I \circ \partial_H\big(C_{k+1}(\tilde{d}^{u+1}_{k+1} * \tilde{d}^{m+1}_{k+2})\big) \end{aligned}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{73}$$

Thus we find that the first two equations of (67) are correct for $u+1$ for our chosen $k \geq 0$. To prove that the other two are also correct we will consider the following guarded recursive specification E_5.

$$\begin{aligned} E_5 = \big\{\, X^5_{m+1} &= t \cdot Y^5_{m+1} + Z^5_1 + Z^5_2 \\ Y^5_m &= c_2(\tilde{d}_{k+2}) \cdot X^5_{m+1} + Z^5_2 \\ Z^5_1 &= c_4(\overline{k}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ Z^5_2 &= l \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \;\big|\; m \geq 0 \,\big\}. \end{aligned}$$

Let $C_5 = \{X^5_{m+1}, Y^5_m : m \geq 0\}$. It is evident that C_5 is a conservative cluster of E_5 in I. The set of exits $U(C_5) = \{Z^5_1, Z^5_2\}$ is finite so in accordance with $CFAR^\infty$ we find for all $m \geq 0$

$$\tau_I(X^5_{m+1}) = \tau_I(Y^5_m) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Let $\beta_5 = \beta\big(I, \{c_4(\overline{k}), l\}\big)$. Then it is easy to see with the aid of equations (54), (61), the induction hypothesis, and the just derived result (73) that if we put for all $m \geq 0$

$$\begin{aligned} X^5_{m+1} &= \pi^p_J \circ \beta_5 \circ \partial_H\big(G_k(\tilde{d}^{u+1}_{k+1} * \tilde{d}^{m+1}_{k+2})\big) \\ Y^5_m &= \pi^p_J \circ \beta_5 \circ \partial_H\big(J_k(\tilde{d}^{u+1}_{k+1} * \tilde{d}^m_{k+2})\big), \end{aligned}$$

this will be a solution for the system E_5. It is deduced in exactly the same way as we did hereinbefore that

$$\left.\begin{array}{l}\pi_J^p \circ \tau_I \circ \partial_H\big(G_k(\tilde{d}_{k+1}^{u+1} * \tilde{d}_{k+2}^{m+1})\big)\\ \pi_J^p \circ \tau_I \circ \partial_H\big(J_k(\tilde{d}_{k+1}^{u+1} * \tilde{d}_{k+2}^{m})\big)\end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Hence, we see that also the second two equations of (67) are valid for $u+1$, so this will end the induction step for our intermediate result since $k \geq 0$ was arbitrarily chosen.

At this point we will state three more guarded recursive specifications before we can conclude the proof of equation (66). The first matter that we are going to discuss is a result on the I-nodes. We want to calculate that for all $n \geq 0$

$$\pi_J^p \circ \tau_I \circ \partial_H\big(I_k(\tilde{d}_{k+1}^{n})\big) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{74}$$

Therefore, we will consider the guarded recursive specification E_6.

$$\begin{aligned} E_6 = \Big\{\ X_0^6 &= c_1(d_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ X_n^6 &= l \cdot X_{n-1}^6 + X_0^6 \ \Big|\ n \geq 1 \Big\}. \end{aligned}$$

Let $C_6 = \{X_n^6 : n \geq 1\}$. The set of exits is $U(C_6) = \{X_0^6\}$. It will be clear that C_6 is a conservative cluster for E_6 in I. Hence, according to $CFAR^\infty$ we find for all $n \geq 1$, which can be obtained for $n = 0$ directly:

$$\tau_I(X_0^6) = \tau_I(X_n^6) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Let $\beta_6 = \beta\big(I, \{c_1(d_{k+2})\}\big)$. We will show that if we put for all $n \geq 0$

$$X_n^6 = \pi_J^p \circ \beta_6 \circ \partial_H\big(I_k(\tilde{d}_{k+1}^{n})\big), \tag{75}$$

this will solve E_6. First, let $n = 0$ and consider the following.

$$\begin{aligned} \pi_J^p \circ \beta_6 \circ \partial_H\big(I_k(\lambda)\big) &\overset{(57)}{=} c_1(d_{k+2}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(J_k(\lambda)\big) \\ &\overset{(67)}{=} c_1(d_{k+2}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

Now we will handle the case $n \geq 1$.

$$\begin{aligned} \pi_J^p \circ \beta_6 \circ \partial_H\big(I_k(\tilde{d}_{k+1}^{n})\big) &\overset{(58)}{=} l \cdot \pi_J^p \circ \beta_6 \circ \partial_H\big(I_k(\tilde{d}_{k+1}^{n-1})\big) \\ &\qquad + c_1(d_{k+2}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(J_k(\tilde{d}_{k+1}^{n})\big) \\ &\overset{(67)}{=} l \cdot \pi_J^p \circ \beta_6 \circ \partial_H\big(I_k(\tilde{d}_{k+1}^{n-1})\big) \\ &\qquad + c_1(d_{k+2}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

So we see that (75) solves the guarded recursive specification E_6. Together with our calculations above we find that equation (74) is valid for all $n \geq 0$. The next thing that we are going to prove is something about the A-nodes. We will show that for all $n \geq 0$

$$\pi_J^p \circ \tau_I \circ \partial_H\big(A_{k+1}(\tilde{d}_{k+1}^{n})\big) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{76}$$

Consider the guarded recursive specification E_7 below.

$$\begin{aligned}
E_7 = \{\ X_0^7 &= c_1(d_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\
X_n^7 &= l \cdot X_{n-1}^7 + X_0^7 + Z_7 \\
Z_7 &= c_3(\tilde{d}_{k+1}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \mid n \geq 1\ \}.
\end{aligned}$$

Let $C_7 = \{X_n^7 : n \geq 1\}$. The set of exits is $U(C_7) = \{X_0^7, Z_7\}$. Since C_7 is a conservative cluster for E_7 in I and since the set of exits is finite, we may apply *CFAR*$^\infty$. This yields for all $n \geq 1$, which we can immediately infer for X_0^7:

$$\tau_I(X_0^7) = \tau_I(X_n^7) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Let $\beta_7 = \beta\big(I, \{c_1(d_{k+2}), c_3(\tilde{d}_{k+1})\}\big)$. We will show that, if we put for all $n \geq 0$

$$X_n^7 = \pi_J^p \circ \beta_7 \circ \partial_H\big(A_{k+1}(\tilde{d}_{k+1}^n)\big),$$

this will solve E_7. First let $n = 0$. We find easily

$$\begin{aligned}
\pi_J^p \circ \beta_7 \circ \partial_H\big(A_{k+1}(\lambda)\big) &\overset{(38)}{=} c_1(d_{k+2}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\lambda)\big) \\
&\overset{(67)}{=} c_1(d_{k+2}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).
\end{aligned}$$

Now we will handle $n \geq 1$.

$$\begin{aligned}
\pi_J^p \circ \beta_7 \circ \partial_H\big(A_{k+1}(\tilde{d}_{k+1}^n)\big) &\overset{(39)}{=} l \cdot \pi_J^p \circ \beta_7 \circ \partial_H\big(A_{k+1}(\tilde{d}_{k+1}^{n-1})\big) \\
&\quad + c_1(d_{k+2}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(B_{k+1}(\tilde{d}_{k+1}^{n-1})\big) \\
&\quad + c_3(\tilde{d}_{k+1}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(I_k(\tilde{d}_{k+1}^{n-1})\big) \\
&\overset{(67)}{=} l \cdot \pi_J^p \circ \beta_7 \circ \partial_H\big(A_{k+1}(\tilde{d}_{k+1}^{n-1})\big) + X_0^7 \\
&\quad + c_3(\tilde{d}_{k+1}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(I_k(\tilde{d}_{k+1}^{n-1})\big) \\
&\overset{(74)}{=} l \cdot \pi_J^p \circ \beta_7 \circ \partial_H\big(A_{k+1}(\tilde{d}_{k+1}^{n-1})\big) + X_0^7 + Z_7
\end{aligned}$$

Combining the just derived results we find with a commutativity/absorption argument that equation (76) is correct. Now we are in a position to prove that equation (66) is valid. We need yet another guarded recursive specification E_8 for that purpose.

$$\begin{aligned}
E_8 = \{\ X_0^8 &= t \cdot Y_0^8 + Z_8 \\
Y_0^8 &= c_2(\tilde{d}_{k+1}) \cdot X_1^8 \\
X_n^8 &= l \cdot X_{n-1}^8 + t \cdot Y_n^8 + Z_8 \\
Y_n^8 &= l \cdot Y_{n-1}^8 + c_2(\tilde{d}_{k+1}) \cdot X_{n+1}^8 \\
Z_8 &= c_4(\overline{k}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \mid n \geq 1\ \}.
\end{aligned}$$

Let $C_8 = \{X_n^8, Y_n^8 : n \geq 0\}$. The set of exits is $U(C_8) = \{Z_8\}$. It is clear that C_8 is conservative, so we apply *CFAR*$^\infty$ and we find for all $n \geq 0$

$$\tau_I(X_n^8) = \tau_I(Y_n^8) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

We will not need a new β. We will take $\beta_3 = \beta\big(I, \{c_4(\overline{k})\}\big)$. Now it will not be difficult to show that, if we put for all $n \geq 0$

$$\begin{aligned} X_n^8 &= \pi_J^p \circ \beta_3 \circ \partial_H\big(F_k(\tilde{d}_{k+1}^n)\big) \\ Y_n^8 &= \pi_J^p \circ \beta_3 \circ \partial_H\big(H_k(\tilde{d}_{k+1}^n)\big), \end{aligned}$$

this will solve E_8. Use equations (50), (51), (55), and (56) together with the just inferred result on the A-nodes (76). Combining the results with an easy commutativity/absorption argument gives that equation (66) is correct if we assume that equations (62)–(65) are valid.

Now we will prove that equation (66) itself is correct. We will do this with induction on p. So let $p = 1$. We will show that the assumptions that we made in equations (62)–(65) are valid for $p = 1$. Let us first take a look at the first assumption (62) and consider the following simple calculation.

$$\pi_J^1 \circ \beta_1 \circ \partial_H\big(D_k(\lambda)\big) \overset{(46)}{=} t \cdot \pi_J^1 \circ \beta_1 \circ \partial_H\big(E_k(\lambda)\big) + s_5(d_{k+1}).$$

So we see that this is just the assumption (62) for $p = 1$. Now let us verify that the second assumption (63) is also valid for $p = 1$.

$$\begin{aligned} \pi_J^1 \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^n)\big) \overset{(47)}{=} & \, l \cdot \pi_J^1 \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^{n-1})\big) \\ & + t \cdot \pi_J^1 \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) + s_5(d_{k+1}). \end{aligned}$$

This is precisely equation (63) for $p = 1$. The verification that the other two assumptions are correct for $p = 1$ can be proved analogously. So we obtain that equation (66) is correct for $p = 1$. This concludes the basis of our induction on p. Now assume that equation (66) is valid for one $p \geq 1$. We will show that the assumptions that we made are now valid for $p + 1$. First, we will verify that equation (62) is correct for $p + 1$. Consider thereunto the following.

$$\begin{aligned} \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(D_k(\lambda)\big) \overset{(46)}{=} & \, t \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(E_k(\lambda)\big) \\ & + s_5(d_{k+1}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(F_k(\lambda)\big) \\ = & \, t \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(E_k(\lambda)\big) \\ & + s_5(d_{k+1}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ = & \, t \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(E_k(\lambda)\big) \\ & + \prod_{i=1}^{p+1} s_5(d_{k+i}). \end{aligned}$$

We see that this is indeed equation (62) for $p + 1$. Now let us make a similar calculation for the second assumption.

$$\begin{aligned} \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(\big(D_k(\tilde{d}_{k+1}^n)\big) \overset{(47)}{=} & \, l \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^{n-1})\big) \\ & + t \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) \\ & + s_5(d_{k+1}) \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(F_k(\tilde{d}_{k+1}^n)\big) \\ = & \, l \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^{n-1})\big) \\ & + t \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) \\ & + s_5(d_{k+1}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ = & \, l \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^{n-1})\big) \\ & + t \cdot \pi_J^{p+1} \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big) \\ & + \prod_{i=1}^{p+1} s_5(d_{k+i}). \end{aligned}$$

We see that this is exactly the second assumption (63) for $p+1$. The proof that the assumptions (64) and (65) are also correct for $p+1$ can be calculated in the same way. But then we can immediately conclude that (66) is correct for $p+1$. This ends our induction on p and we find that equation (66) is correct for all p. With this knowledge it is evident that the assumptions that we made are also correct by themselves. (For, the case $p = 1$ was already correct and for $p > 1$ we can reiterate, using (66), the above calculation.) We will use this fact to prove that the following holds for all $n \geq 0$.

$$\left.\begin{array}{l}\pi_J^p \circ \tau_I \circ \partial_H\big(D_k(\tilde{d}_{k+1}^n)\big)\\ \pi_J^p \circ \tau_I \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big)\end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+i}). \tag{77}$$

Consider the subsequent guarded recursive specification E_9.

$$\begin{aligned} E_9 = \{\, X_0^9 &= t \cdot Y_0^9 + Z_9 \\ Y_0^9 &= c_2(\tilde{d}_{k+1}) \cdot X_1^9 + Z_9 \\ X_n^9 &= l \cdot X_{n-1}^9 + t \cdot Y_n^9 + Z_9 \\ Y_n^9 &= l \cdot Y_{n-1}^9 + c_2(\tilde{d}_{k+1}) \cdot X_{n+1}^9 + Z_9 \\ Z_9 &= \prod_{i=1}^{p} s_5(d_{k+i}) \;\big|\; n \geq 1 \,\}. \end{aligned}$$

We see at once from equations (62)–(65) that if we put for all $n \geq 0$

$$\begin{aligned} X_n^9 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(D_k(\tilde{d}_{k+1}^n)\big) \\ Y_n^9 &= \pi_J^p \circ \beta_1 \circ \partial_H\big(E_k(\tilde{d}_{k+1}^n)\big), \end{aligned}$$

that this will solve the system E_9. Let $C_9 = \{X_n^9, Y_n^9 : n \geq 0\}$. The set of exits is $U(C_9) = \{Z_9\}$. We may apply $CFAR^\infty$ and we find for all $n \geq 0$

$$\tau_I(X_n^9) = \tau_I(Y_n^9) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+i}).$$

A simple commutativity/absorption argument will yield that equation (77) is valid. In the sequel we will need the following result for a single *BC*-trace. For all $n \geq 0$

$$\left.\begin{array}{l}\pi_J^p \circ \tau_I \circ \partial_H\big(B_k(\tilde{d}_{k+1}^n)\big)\\ \pi_J^p \circ \tau_I \circ \partial_H\big(C_k(\tilde{d}_{k+1}^n)\big)\end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+i}). \tag{78}$$

To achieve this, we will need one more guarded recursive specification E_{10}.

$$\begin{aligned} E_{10} = \{\, X_0^{10} &= c_2(\tilde{d}_{k+1}) \cdot Y_1^{10} \\ Y_0^{10} &= t \cdot X_0^{10} \\ X_n^{10} &= l \cdot X_{n-1}^{10} + c_2(\tilde{d}_{k+1}) \cdot Y_{n+1}^{10} + Z_{10} \\ Y_n^{10} &= l \cdot Y_{n-1}^{10} + t \cdot X_n^{10} + Z_{10} \\ Z_{10} &= c_3(\tilde{d}_{k+1}) \cdot \prod_{i=1}^{p} s_5(d_{k+i}) \;\big|\; n \geq 1 \,\}. \end{aligned}$$

Let $C_{10} = \{X_n^{10}, Y_n^{10} : n \geq 0\}$. It will be clear that this is a conservative cluster for E_{10} in I. The set of exits is $U(C_{10}) = \{Z_{10}\}$. So if we apply $CFAR^\infty$ we find for all $n \geq 0$

$$\tau_I(X_n^{10}) = \tau_I(Y_n^{10}) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+i}).$$

Let $\beta_8 = \beta\big(I, \{c_3(\tilde{d}_{k+1})\}\big)$. With the aid of equations (40), (41), (43) and (44) together with the just derived result on the *DE*-traces (77) it is not hard to see that if we put for all $n \geq 0$

$$\begin{aligned} X_n^{10} &= \pi_J^p \circ \beta_8 \circ \partial_H\big(B_k(\tilde{d}_{k+1}^n)\big) \\ Y_n^{10} &= \pi_J^p \circ \beta_8 \circ \partial_H\big(C_k(\tilde{d}_{k+1}^n)\big), \end{aligned}$$

this is a solution for the guarded recursive specification E_{10}. So a straightforward calculation gives us that equation (78) is correct.

Hereinafter, we will give the final calculation in order to prove (4.14). Choose $k = 0$ and consider the following.

$$\begin{aligned} \pi_J^p \circ \tau_I \circ \partial_H\big(A_0(\lambda)\big) &\overset{(38)}{=} \tau \cdot \pi_J^p \circ \tau_I \circ \partial_H\big(B_0(\lambda)\big) \\ &\overset{(78)}{=} \tau \cdot \prod_{i=1}^{p} s_5(d_i) \\ &= \tau \cdot \pi_J^p\Big(\prod_{i=1}^{\infty} s_5(d_i)\Big). \end{aligned}$$

So we see that we have shown for a fixed $p \geq 1$

$$\pi_J^p \circ \tau_I \circ \partial_H\Big(S(\overline{0}) \parallel Q(\lambda) \parallel R(\overline{0}) \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big) = \tau \cdot \pi_J^p\Big(\prod_{i=1}^{\infty} s_5(d_i)\Big).$$

With the use of *GIP* we conclude the proof of (4.14).

In the following application we will increase the unreliability of the queue. We will allow that the queue can lose or duplicate any datum contained in it at any moment. In order to be able to give a guarded recursive specification for this faulty and stuttering queue, we will need the notion of a residue set and a stuttering set of the words that the faulty and stuttering queue can contain. The notion of a residue set is taken from [**11**].

Definitions (4.15)

Let $n \geq 1$ and let $\sigma \in E^*$ be a non-empty word, say $\sigma = e_1 * e_2 * \cdots * e_n$. For $n \geq 2$ the residue set $R(\sigma)$ of σ is the following set

$$R(\sigma) = \{e_1 * \cdots * e_{n-1}, e_2 * \cdots * e_n\} \cup \{e_1 * \cdots * e_{i-1} * e_{i+1} * \cdots * e_n : 1 < i < n\}$$

and if $n = 1$ we will have for the residue set $R(e_1) = \{\lambda\}$. For the stuttering set we will not need this dichotomy. The stuttering set $S(\sigma)$ of σ is the following set

$$S(\sigma) = \{e_1 * \cdots * e_i * e_i * \cdots * e_n : 1 \leq i \leq n\}.$$

Just as with the previous queue we will express the losing behaviour with an atomic action l. The stuttering will be expressed with an atomic action s. Now we are in a position to give the guarded recursive specification for the faulty and stuttering queue.

$$\begin{aligned} Q(\lambda) &= \sum_{x \in E} r_2(x) \cdot Q(x) \\ Q(x * \sigma) &= \sum_{y \in E} r_2(y) \cdot Q(x * \sigma * y) + s_3(x) \cdot Q(\sigma) \\ &\quad + \sum_{\rho_1 \in R(x*\sigma)} l \cdot Q(\rho_1) + \sum_{\rho_2 \in S(x*\sigma)} s \cdot Q(\rho_2). \end{aligned}$$

The other components of our alternating bit protocol will remain the same just as in the former theorem. We will reiterate the encapsulation set H and we will give a new abstraction set I that only differs from the previous one in having an extra s.

$$\begin{aligned} H = &\left\{ r_1(d), s_1(d) : d \in D \right\} \\ &\cup \left\{ r_2(x), r_3(x), s_2(x), s_3(x) : x \in E \right\} \\ &\cup \left\{ r_4(\overline{k}), s_4(\overline{k}) : \overline{k} \in \mathbb{Z}/2\mathbb{Z} \right\} \\ I = &\left\{ c_1(d) : d \in D \right\} \\ &\cup \left\{ c_2(x), c_3(x) : x \in E \right\} \\ &\cup \left\{ c_4(\overline{k}) : \overline{k} \in \mathbb{Z}/2\mathbb{Z} \right\} \cup \{l, t, s\}. \end{aligned}$$

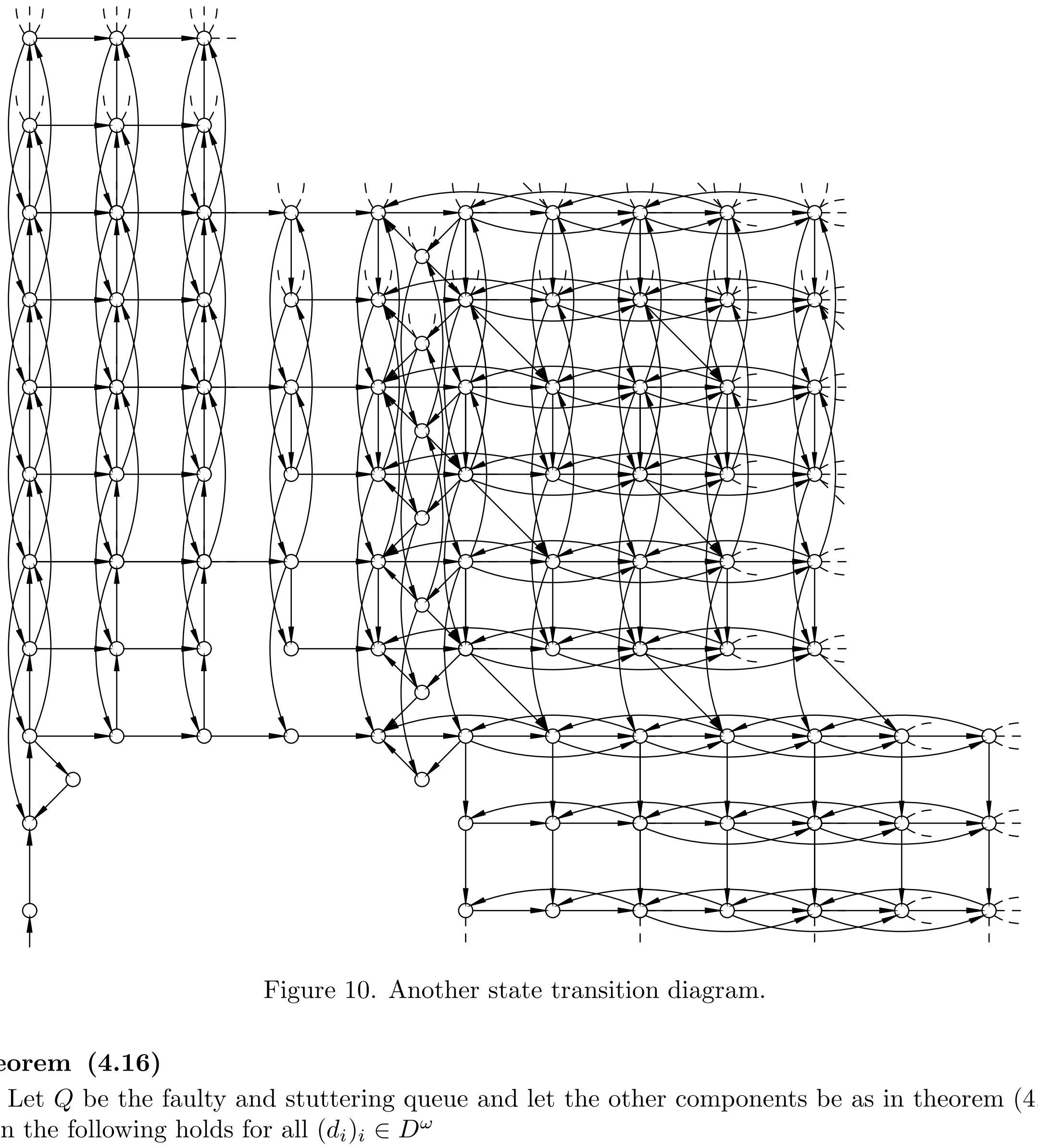

Figure 10. Another state transition diagram.

Theorem (4.16)

Let Q be the faulty and stuttering queue and let the other components be as in theorem (4.8). Then the following holds for all $(d_i)_i \in D^\omega$

$$\tau_I \circ \partial_H \Big(S(\overline{0}) \parallel Q(\lambda) \parallel R(\overline{0}) \parallel \prod_{i=1}^{\infty} s_1(d_i) \Big) = \tau \cdot \prod_{i=1}^{\infty} s_5(d_i).$$

Proof. The structure of this proof will be not entirely the same as before. Let $(d_i)_i \in D^\omega$ be fixed. We will first calculate

$$\partial_H\Big(S(\overline{0}) \parallel Q(\lambda) \parallel R(\overline{0}) \parallel \prod_{i=1}^{\infty} s_1(d_i)\Big).$$

We have schematically depicted this process in figure 10 on page 52. We will leave out the actual calculation of this process but we will use figure 10 to provide some explanation on the resulting guarded recursive specification. For the guarded recursive specification that follows we will use the abbreviations $A_k(\sigma), B_k(\sigma), \ldots, J_k(\sigma)$ that we already saw in theorems (4.8) and (4.14). We will alternate pieces of the guarded recursive specification with informal comment on them to optimize the readability of the specification. Moreover, the order of the equations will be different than in the former theorems. It is determined by a walk through figure 10.

We suppose that k, n, m range over the set of natural numbers with $k \geq 0$ and $n, m \geq 1$. Informally, we will start walking at the initial point $\partial_H\big(A_0(\lambda)\big)$ of the graph in figure 10. Formally, we will start for all $k \geq 0$ with equation $\partial_H\big(A_k(\lambda)\big)$. The first equation corresponds to the transition from the initial node to the only node that can be reached from that point: the second node. This equation is the same as it appeared in theorem (4.14).

$$\partial_H\big(A_k(\lambda)\big) = c_1(d_{k+1}) \cdot \partial_H\big(B_k(\lambda)\big).$$

This is not surprising since in this phase the faulty and stuttering queue is empty. In the the second node the queue is still empty so this equation will not differ from the one in the former theorem, too.

$$\partial_H\big(B_k(\lambda)\big) = c_2(\tilde{d}_{k+1}) \cdot \partial_H\big(C_k(\tilde{d}_{k+1})\big).$$

In the third node several things can occur: a time out can be given (we will go up one node in the graph), a loss of any datum can be given (we will go southeastwards and enter the triangle), a duplication of any datum can be performed (we will raise two nodes in the graph with an arched edge) or a communication can be executed (we will go to the right in the graph). This behaviour will be repeated in the node that we enter if the duplication has taken place and so on see the picture. So we obtain for all these nodes the following equation

$$\begin{aligned}\partial_H\big(C_k(\tilde{d}_{k+1}^n)\big) = l \cdot \partial_H\big(C_k(\tilde{d}_{k+1}^{n-1})\big) + s \cdot \partial_H\big(C_k(\tilde{d}_{k+1}^{n+1})\big)\\ + t \cdot \partial_H\big(B_k(\tilde{d}_{k+1}^n)\big) + c_3(\tilde{d}_{k+1}) \cdot \partial_H\big(D_k(\tilde{d}_{k+1}^{n-1})\big).\end{aligned}$$

If $n = 1$ we will enter the triangle. In that node only a time out can be performed and we come back in the second node. This corresponds to the equation

$$\partial_H\big(C_k(\lambda)\big) = t \cdot \partial_H\big(B_k(\lambda)\big).$$

Half of the first trace up is now clarified but we must take a look at the even nodes of this trace. As with the odd nodes four actions can be executed. We can lose any datum element and go down two nodes with an arched edge, a duplication of any datum can be performed and we will go up two nodes with an arched edge, the queue can be loaded with one more datum of the same type (we will go up one node in the graph) or a communication can take place and a datum will be removed from the queue (we will go to the right in the graph to the second trace up). Observe that we are dealing with a queue that contains one type of data, so the residue set and the stuttering set both contain one element so there is one stuttering edge and one losing edge. We will have the following equation for these even nodes

$$\begin{aligned}\partial_H\big(B_k(\tilde{d}_{k+1}^n)\big) = l \cdot \partial_H\big(B_k(\tilde{d}_{k+1}^{n-1})\big) + s \cdot \partial_H\big(B_k(\tilde{d}_{k+1}^{n+1})\big)\\ + c_2(\tilde{d}_{k+1}) \cdot \partial_H\big(C_k(\tilde{d}_{k+1}^{n+1})\big) + c_3(\tilde{d}_{k+1}) \cdot \partial_H\big(E_k(\tilde{d}_{k+1}^{n-1})\big).\end{aligned}$$

This trace up will be called a single *BC*-trace since the queue contains only a single type of elements. We have seen that the only way out from a single *BC*-trace is by entering what we will call a *DE*-trace. We will discuss this second trace. We will start with the lowest node of the second trace. In this phase the queue is empty. A time out can be executed and we will go up one node or a send action at port 5 can take place (we go to the right in the graph). This corresponds to the following equation

$$\partial_H\big(D_k(\lambda)\big) = t \cdot \partial_H\big(E_k(\lambda)\big) + s_5(d_{k+1}) \cdot \partial_H\big(F_k(\lambda)\big).$$

Now we will discuss the second node of the *DE*-trace. At that node we can load the queue with a datum element or we can perform the send action to the right, as well. So we obtain a similar equation.

$$\partial_H\big(E_k(\lambda)\big) = c_2(\tilde{d}_{k+1}) \cdot \partial_H\big(D_k(\tilde{d}_{k+1})\big) + s_5(d_{k+1}) \cdot \partial_H\big(H_k(\lambda)\big).$$

Now we take a node of this trace that is nor the first neither the second. Then we will have two more possible actions that can be performed since the queue it not empty at this stage: an additional duplication or deletion can be carried out. We can see this in the graph. The following two equations will complete the description of the *DE*-trace.

$$\begin{aligned}
\partial_H\big(D_k(\tilde{d}^n_{k+1})\big) &= l \cdot \partial_H\big(D_k(\tilde{d}^{n-1}_{k+1})\big) + s \cdot \partial_H\big(D_k(\tilde{d}^{n+1}_{k+1})\big)\\
&\quad + t \cdot \partial_H\big(E_k(\tilde{d}^n_{k+1})\big) + s_5(d_{k+1}) \cdot \partial_H\big(F_k(\tilde{d}^n_{k+1})\big)\\
\partial_H\big(E_k(\tilde{d}^n_{k+1})\big) &= l \cdot \partial_H\big(E_k(\tilde{d}^{n-1}_{k+1})\big) + s \cdot \partial_H\big(E_k(\tilde{d}^{n+1}_{k+1})\big)\\
&\quad + c_2(\tilde{d}_{k+1}) \cdot \partial_H\big(D_k(\tilde{d}^{n+1}_{k+1})\big) + s_5(d_{k+1}) \cdot \partial_H\big(H_k(\tilde{d}^n_{k+1})\big).
\end{aligned}$$

As with the single *BC*-trace the way out from the *DE*-trace is performing a send action and entering an *FH*-trace. We will start at the lowest level of this trace. The queue is empty and just two actions can be done: a time out can be given and we will go up in the *FH*-trace or a communication can take place that acknowledges the transmission of the send action that was executed before (when leaving the *DE*-trace) and we will enter the subsequent *A*-node (we will go to the right). This corresponds to the following equation

$$\partial_H\big(F_k(\lambda)\big) = t \cdot \partial_H\big(H_k(\lambda)\big) + c_4(\overline{k}) \cdot \partial_H\big(A_{k+1}(\lambda)\big).$$

Now we will discuss the second node of the *FH*-trace. In that phase the queue is still empty, for, we entered it due to a time out. There is just one action that can be executed: the queue will be loaded with a datum element (the one that was just sent away). This will be represented by the following equation

$$\partial_H\big(H_k(\lambda)\big) = c_2(\tilde{d}_{k+1}) \cdot \partial_H\big(F_k(\tilde{d}_{k+1})\big).$$

Taking a node of the *FH*-trace that is nor the first neither the second will yield two more possibilities: duplication or deletion of any datum. We can see this in the picture. Only in the odd nodes we can enter a dense block with entrances that are *A*-nodes. See the following two equations that complete the discussion of the *FH*-traces.

$$\begin{aligned}
\partial_H\big(F_k(\tilde{d}^n_{k+1})\big) &= l \cdot \partial_H\big(F_k(\tilde{d}^{n-1}_{k+1})\big) + s \cdot \partial_H\big(F_k(\tilde{d}^{n+1}_{k+1})\big)\\
&\quad + t \cdot \partial_H\big(H_k(\tilde{d}^n_{k+1})\big) + c_4(\overline{k}) \cdot \partial_H\big(A_{k+1}(\tilde{d}^n_{k+1})\big)\\
\partial_H\big(H_k(\tilde{d}^n_{k+1})\big) &= l \cdot \partial_H\big(H_k(\tilde{d}^{n-1}_{k+1})\big) + s \cdot \partial_H\big(H_k(\tilde{d}^{n+1}_{k+1})\big)\\
&\quad + c_2(\tilde{d}_{k+1}) \cdot \partial_H\big(F_k(\tilde{d}^{n+1}_{k+1})\big).
\end{aligned}$$

The only nodes that we did not discuss yet are the *A*-nodes that contain datum elements that are old: the send action has been executed but the acknowledgement not and due to time outs the queue will be loaded with old data. This can be clearly seen in the description of the *FH*-trace. So we will discuss here the A-nodes with a non-empty queue. In this phase duplication and deletion can take place (these correspond in figure 10 to the up and down going arched edges). Besides this behaviour a new datum element can be taken from the test row (we go to the right in the picture and we enter a B-node) or a communication can take place with the receiver such that one datum can be removed from the queue (we will go down one node and enter a I-node). The equation that accompanies this is as follows.

$$\begin{aligned}\partial_H\big(A_{k+1}(\tilde{d}^n_{k+1})\big) = {} & l\cdot\partial_H\big(A_{k+1}(\tilde{d}^{n-1}_{k+1})\big) + s\cdot\partial_H\big(A_{k+1}(\tilde{d}^{n+1}_{k+1})\big)\\ & + c_1(d_{k+2})\cdot\partial_H\big(B_{k+1}(\tilde{d}^n_{k+1})\big) + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(I_k(\tilde{d}^{n-1}_{k+1})\big).\end{aligned}$$

First, we will treat the I-nodes. From the lowest one we can do just one thing: take the following element of the test row (we go to the right in the picture). On the other levels the queue is not empty so we will also have the possibility to duplicate or to delete (the arched up and down going edges in the figure). We will give both equations for the I-nodes.

$$\begin{aligned}\partial_H\big(I_k(\lambda)\big) &= c_1(d_{k+2})\cdot\partial_H\big(J_k(\lambda)\big)\\ \partial_H\big(I_k(\tilde{d}^n_{k+1})\big) &= l\cdot\partial_H\big(I_k(\tilde{d}^{n-1}_{k+1})\big) + s\cdot\partial_H\big(I_k(\tilde{d}^{n+1}_{k+1})\big)\\ &\quad + c_1(d_{k+2})\cdot\partial_H\big(J_k(\tilde{d}^n_{k+1})\big).\end{aligned}$$

Now we will treat the B-nodes. Recall that they are at the right of the A-nodes. We can lose or duplicate or we can load or communicate. After deletions and stutterings we will still be in an B-node (the arched edges). If we load we will go tho the right in the picture (we will enter the double BC-trace since there are now two types of data) and if we communicate we will go one node down in the graph. We will give the equation of the B-nodes.

$$\begin{aligned}\partial_H\big(B_{k+1}(\tilde{d}^n_{k+1})\big) = {} & l\cdot\partial_H\big(B_{k+1}(\tilde{d}^{n-1}_{k+1})\big) + s\cdot\partial_H\big(B_{k+1}(\tilde{d}^{n+1}_{k+1})\big)\\ & + c_2(\tilde{d}_{k+2})\cdot\partial_H\big(C_{k+1}(\tilde{d}^n_{k+2} * \tilde{d}_{k+2})\big) + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(J_k(\tilde{d}^{n-1}_{k+1})\big).\end{aligned}$$

Observe that we have in the node that is right from this B-node two types of data in the queue: old data and a new one. Since we can lose or duplicate any datum that is in the queue at any moment we will see in the graph more edges than in the first part of the figure: we have two deletions and two duplications. The triangles that we can see in the picture are due to the fact that the just read new datum can be deleted immediately. This means that there are also C-nodes. First, we will discuss the node with the old and new data. As we already said there are two types of deletion and duplication (arched edges to the right and two nodes up are the stuttering edges; an arched edge two nodes down and the small edge to the triangle node are the deletion edges). Also a time out can be given (we will go to the right one node in the graph). Finally, a communication with the receiver can be executed and we will go down one node in the graph. This is expressed in the following equation (recall that $n, m \geq 1$).

$$\begin{aligned}\partial_H\big(C_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big) = {} & l\cdot\partial_H\big(C_{k+1}(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2})\big) + l\cdot\partial_H\big(C_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m-1}_{k+2})\big)\\ & + s\cdot\partial_H\big(C_{k+1}(\tilde{d}^{n+1}_{k+1} * \tilde{d}^m_{k+2})\big) + s\cdot\partial_H\big(C_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m+1}_{k+2})\big)\\ & + t\cdot\partial_H\big(B_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big) + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(G_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2})\big).\end{aligned}$$

Now we will discuss the C-nodes. In these nodes there is only one type of data left: the old ones. So we will have here one type of deletion and duplication (the arched edges up and down at the triangle nodes). A time out can be performed (we will go northwestwards in the triangle and return to the B-node) or we can communicate with the receiver and an old datum can be removed from the queue (we will go southeastwards in the graph). We will give the corresponding equation in the next display.

$$\begin{aligned}\partial_H\big(C_{k+1}(\tilde{d}^n_{k+1})\big) = {} & l\cdot\partial_H\big(C_{k+1}(\tilde{d}^{n-1}_{k+1})\big) + s\cdot\partial_H\big(C_{k+1}(\tilde{d}^{n+1}_{k+1})\big)\\ & + t\cdot\partial_H\big(B_{k+1}(\tilde{d}^n_{k+1})\big) + c_3(\tilde{d}_{k+1})\cdot\partial_H\big(G_k(\tilde{d}^{n-1}_{k+1})\big).\end{aligned}$$

We will finish the discussion of the double BC-trace by giving the equation for the B-nodes that contain two types of data. Just as with the C-nodes of this trace we will have two types of deletion and duplication (for the stuttering we go up and to the right with an arched edge and for the deletion we will go down and to the left with an arched edge). Besides these things we can also stay in the double BC-trace by loading one more new datum in the queue (we go one node to the right) or we can escape it by communicating with the receiver (we go one node down in the picture). We will give the equation for all these nodes.

$$\begin{aligned}\partial_H\big(B_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big) = {} & l \cdot \partial_H\big(B_{k+1}(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2})\big) + l \cdot \partial_H\big(B_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m-1}_{k+2})\big)\\ & + s \cdot \partial_H\big(B_{k+1}(\tilde{d}^{n+1}_{k+1} * \tilde{d}^m_{k+2})\big) + s \cdot \partial_H\big(B_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m+1}_{k+2})\big)\\ & + c_2(\tilde{d}_{k+2}) \cdot \partial_H\big(C_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m+1}_{k+2})\big)\\ & + c_3(\tilde{d}_{k+1}) \cdot \partial_H\big(J_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2})\big).\end{aligned}$$

To escape a single BC-trace we communicate with the receiver and we enter a DE-trace. With the double BC-trace we communicate with the receiver and we will enter the JG-trace. These traces are the only ones that remain for our discussion. For both nodes we will have four equations. First, we will treat the nodes with an empty queue. The location of the empty G-node in the picture is the lowest node of the dense block; it is part of a triangle. If we go from that point northwestwardly to the next node we will enter the empty J-node. In this node the only possible action is the loading of the queue with a new datum element (we will go to the right in the picture). This is expressed in an equation as follows.

$$\partial_H\big(J_k(\lambda)\big) = c_2(\tilde{d}_{k+2}) \cdot \partial_H\big(G_k(\tilde{d}_{k+2})\big).$$

We see that in this equation there is a G-node with only a new datum. We will discuss this node after the empty G-node. In this node we have two possible actions. We can perform a time out (we go northwestwardly to the empty J-node) or an acknowledgement can be executed and we enter the beginning of the next single BC-trace. This is the corresponding equation

$$\partial_H\big(G_k(\lambda)\big) = t \cdot \partial_H\big(J_k(\lambda)\big) + c_4(\overline{k}) \cdot \partial_H\big(B_{k+1}(\lambda)\big).$$

The lowest horizontal line of the dense block is a somewhat different JG-trace than the others: in this trace all the old datum elements are removed from the queue and only new data is in the queue. Therefore, there is one type of deletion and duplication (we go to the left or to the right with arched edges). At a G-node we can execute a time out and we stay in the JG-trace (we go one node to the right in the figure) or we can perform an acknowledgement and we go to the next single BC-trace (we go to the southeast in the graph). For the J-nodes there is besides the stuttering and deletion actions just one possible action: the queue will be loaded with a new datum element (we will go one node to the right in the graph). We will give both equations hereinafter.

$$\begin{aligned}\partial_H\big(G_k(\tilde{d}^m_{k+2})\big) &= l \cdot \partial_H\big(G_k(\tilde{d}^{m-1}_{k+2})\big) + s \cdot \partial_H\big(G_k(\tilde{d}^{m+1}_{k+2})\big)\\ &\quad + t \cdot \partial_H\big(J_k(\tilde{d}^m_{k+2})\big) + c_4(\overline{k}) \cdot \partial_H\big(B_{k+1}(\tilde{d}^m_{k+2})\big)\\ \partial_H\big(J_k(\tilde{d}^m_{k+2})\big) &= l \cdot \partial_H\big(J_k(\tilde{d}^{m-1}_{k+2})\big) + s \cdot \partial_H\big(J_k(\tilde{d}^{m+1}_{k+2})\big)\\ &\quad + c_2(\tilde{d}_{k+2}) \cdot \partial_H\big(G_k(\tilde{d}^{m+1}_{k+2})\big).\end{aligned}$$

The JG-traces have at their beginning a triangle as with the beginning of the double BC-traces. In fact, we have in this dense block a horizontal CG-trace. The C-nodes of this trace have already been treated. Now we will handle the other part of this CG-trace. Since the empty G-node is already discussed we will treat the non-empty ones. In these nodes there are only old datum elements. Hence, we will have one deletion and one duplication (see the up and down going arched edges). A time out can be executed (we will go northwestwardly) and we return in the first J-node of the JG-trace or an acknowledgement is performed (we will go soutwestwards in the graph). We will give the equation.

$$\begin{aligned}\partial_H\big(G_k(\tilde{d}^n_{k+1})\big) &= l \cdot \partial_H\big(G_k(\tilde{d}^{n-1}_{k+1})\big) + s \cdot \partial_H\big(G_k(\tilde{d}^{n+1}_{k+1})\big)\\ &\quad + t \cdot \partial_H\big(J_k(\tilde{d}^n_{k+1})\big) + c_4(\overline{k}) \cdot \partial_H\big(B_{k+1}(\tilde{d}^n_{k+1})\big).\end{aligned}$$

We see in this equation that also a J-node can contain only old data. We will discuss them hereinafter. Besides losing and stuttering (we will go up and down with arched edges) the queue can be loaded with a new datum element (we go one node to the right in the picture). We will give this equation below.

$$\begin{aligned}\partial_H\big(J_k(\tilde{d}^n_{k+1})\big) = l\cdot\partial_H\big(J_k(\tilde{d}^{n-1}_{k+1})\big) + s\cdot\partial_H\big(J_k(\tilde{d}^{n+1}_{k+1})\big)\\ + c_2(\tilde{d}_{k+2})\cdot\partial_H\big(G_k(\tilde{d}^n_{k+1}*\tilde{d}_{k+2})\big).\end{aligned}$$

The two remaining equations concern the JG-traces with two types of datum elements. For the G-nodes we have beside the four stuttering and deletion actions (see the four arched edges in the graph) two possible actions: we can stay in the JG-trace and perform a time out (we go one node to the right in the graph) or an acknowledgement is executed (we will go southeastwards to the lower double BC-trace). We will give its equation.

$$\begin{aligned}\partial_H\big(G_k(\tilde{d}^n_{k+1}*\tilde{d}^m_{k+2})\big) &= l\cdot\partial_H\big(G_k(\tilde{d}^{n-1}_{k+1}*\tilde{d}^m_{k+2})\big) + l\cdot\partial_H\big(G_k(\tilde{d}^n_{k+1}*\tilde{d}^{m-1}_{k+2})\big)\\ &+ s\cdot\partial_H\big(G_k(\tilde{d}^{n+1}_{k+1}*\tilde{d}^m_{k+2})\big) + s\cdot\partial_H\big(G_k(\tilde{d}^n_{k+1}*\tilde{d}^{m+1}_{k+2})\big)\\ &+ t\cdot\partial_H\big(J_k(\tilde{d}^n_{k+1}*\tilde{d}^m_{k+2})\big) + c_4(\overline{k})\cdot\partial_H\big(B_{k+1}(\tilde{d}^n_{k+1}*\tilde{d}^m_{k+2})\big).\end{aligned}$$

Finally, for the J-nodes with two types of data we have apart from the four obvious possibilities (see the arched edges in the graph) just one other action: the queue will be loaded with a new datum element (we go one node to the right in the picture). We will give the equation for these nodes.

$$\begin{aligned}\partial_H\big(J_k(\tilde{d}^n_{k+1}*\tilde{d}^m_{k+2})\big) &= l\cdot\partial_H\big(J_k(\tilde{d}^{n-1}_{k+1}*\tilde{d}^m_{k+2})\big) + l\cdot\partial_H\big(J_k(\tilde{d}^n_{k+1}*\tilde{d}^{m-1}_{k+2})\big)\\ &+ s\cdot\partial_H\big(J_k(\tilde{d}^{n+1}_{k+1}*\tilde{d}^m_{k+2})\big) + s\cdot\partial_H\big(J_k(\tilde{d}^n_{k+1}*\tilde{d}^{m+1}_{k+2})\big)\\ &+ c_2(\tilde{d}_{k+2})\cdot\partial_H\big(G_k(\tilde{d}^n_{k+1}*\tilde{d}^{m+1}_{k+2})\big).\end{aligned}$$

In the former theorems we used the big guarded recursive specification in the sequel of the proof. If one tries to imitate the proofs that have been exposed thus far, it will turn out that there is a problem with the dense block of figure 10: in the former verifications these blocks have been teared down with induction on the “floor height”. We can see in figure 9 on page 40 that there are no arrows that go up from a floor; they all go to the right, down or to the southeast. In the dense block of figure 10 there are arrows that go up from all but the lowest floor. This is due to the duplication of old data. If we assume that only the last read datum element can be duplicated then we can use the above deduced guarded recursive specification. But now we can not use this induction argument since we can reach all the floors above us. What we will use is the fact that this dense block not just looks like a cluster: it can be made into a cluster in the sense of definition (4.10). To achieve the desired situation we have to introduce a new guarded recursive specification that is heavily based on the one that we found above. We will write down an acuted version of the above guarded recursive specification in which we will also acute some of the atomic actions that appear in the dense block. All the atomic actions in the dense block that result in a transition such that the queue becomes empty or does not contain old datum elements anymore will be acuted. Before we will explain more of the structure of this proof we will enumerate the guarded recursive specification and thereinafter we will show that we may use this specification in order to prove (4.16). Let k, n, m range over the set of natural numbers with $k \geq 0$, $n, m \geq 1$. In the subsequent display we will asume that $k \geq 0$ and $m \geq 1$ in all the equations. For $n \geq 1$ we will mention explicitly what the range is.

$$A'_k(\lambda) = c_1(d_{k+1})\cdot B'_k(\lambda) \tag{79}$$

$$\begin{aligned}A'_{k+1}(\tilde{d}^n_{k+1}) &= l\cdot A'_{k+1}(\tilde{d}^{n-1}_{k+1}) + s\cdot A'_{k+1}(\tilde{d}^{n+1}_{k+1}) \qquad (n\geq 1)\\ &+ c_1(d_{k+2})\cdot B'_{k+1}(\tilde{d}^n_{k+1}) + c_3(\tilde{d}_{k+1})\cdot I'_k(\tilde{d}^{n-1}_{k+1})\end{aligned} \tag{80}$$

$$\begin{aligned}
B'_k(\lambda) &= c_2(\tilde{d}_{k+1}) \cdot C'_k(\tilde{d}_{k+1}) && (81)\\
B'_k(\tilde{d}^n_{k+1}) &= l \cdot B'_k(\tilde{d}^{n-1}_{k+1}) + s \cdot B'_k(\tilde{d}^{n+1}_{k+1}) && (n \geq 1)\\
&\quad + c_2(\tilde{d}_{k+1}) \cdot C'_k(\tilde{d}^{n+1}_{k+1}) + c_3(\tilde{d}_{k+1}) \cdot E'_k(\tilde{d}^{n-1}_{k+1}) && (82)\\
B'_{k+1}(\tilde{d}_{k+1}) &= l' \cdot B'_{k+1}(\lambda) + s \cdot B'_{k+1}(\tilde{d}^2_{k+1})\\
&\quad + c_2(\tilde{d}_{k+2}) \cdot C'_{k+1}(\tilde{d}_{k+1} * \tilde{d}_{k+2}) + c'_3(\tilde{d}_{k+1}) \cdot J'_k(\lambda) && (83)\\
B'_{k+1}(\tilde{d}^n_{k+1}) &= l \cdot B'_{k+1}(\tilde{d}^{n-1}_{k+1}) + s \cdot B'_{k+1}(\tilde{d}^{n+1}_{k+1}) && (n \geq 2)\\
&\quad + c_2(\tilde{d}_{k+2}) \cdot C'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}_{k+2}) + c_3(\tilde{d}_{k+1}) \cdot J'_k(\tilde{d}^{n-1}_{k+1}) && (84)\\
B'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^m_{k+2}) &= l' \cdot B'_{k+1}(\tilde{d}^m_{k+2}) + l \cdot B'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^{m-1}_{k+2})\\
&\quad + s \cdot B'_{k+1}(\tilde{d}^2_{k+1} * \tilde{d}^m_{k+2}) + s \cdot B'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^{m+1}_{k+2})\\
&\quad + c_2(\tilde{d}_{k+2}) \cdot C'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^{m+1}_{k+2}) + c'_3(\tilde{d}_{k+1}) \cdot J'_k(\tilde{d}^m_{k+2}) && (85)\\
B'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2}) &= l \cdot B'_{k+1}(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2}) + l \cdot B'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m-1}_{k+2}) && (n \geq 2)\\
&\quad + s \cdot B'_{k+1}(\tilde{d}^{n+1}_{k+1} * \tilde{d}^m_{k+2}) + s \cdot B'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m+1}_{k+2})\\
&\quad + c_2(\tilde{d}_{k+2}) \cdot C'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m+1}_{k+2}) + c_3(\tilde{d}_{k+1}) \cdot J'_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2}) && (86)
\end{aligned}$$

$$\begin{aligned}
C'_k(\lambda) &= t \cdot B'_k(\lambda) && (87)\\
C'_k(\tilde{d}^n_{k+1}) &= l \cdot C'_k(\tilde{d}^{n-1}_{k+1}) + s \cdot C'_k(\tilde{d}^{n+1}_{k+1}) && (n \geq 1)\\
&\quad + t \cdot B'_k(\tilde{d}^n_{k+1}) + c_3(\tilde{d}_{k+1}) \cdot D'_k(\tilde{d}^{n-1}_{k+1}) && (88)\\
C'_{k+1}(\tilde{d}_{k+1}) &= l' \cdot C'_{k+1}(\lambda) + s \cdot C'_{k+1}(\tilde{d}^2_{k+1})\\
&\quad + t \cdot B'_{k+1}(\tilde{d}_{k+1}) + c'_3(\tilde{d}_{k+1}) \cdot G'_k(\lambda) && (89)\\
C'_{k+1}(\tilde{d}^n_{k+1}) &= l \cdot C'_{k+1}(\tilde{d}^{n-1}_{k+1}) + s \cdot C'_{k+1}(\tilde{d}^{n+1}_{k+1}) && (n \geq 2)\\
&\quad + t \cdot B'_{k+1}(\tilde{d}^n_{k+1}) + c_3(\tilde{d}_{k+1}) \cdot G'_k(\tilde{d}^{n-1}_{k+1}) && (90)\\
C'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^m_{k+2}) &= l' \cdot C'_{k+1}(\tilde{d}^m_{k+2}) + l \cdot C'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^{m-1}_{k+2})\\
&\quad + s \cdot C'_{k+1}(\tilde{d}^2_{k+1} * \tilde{d}^m_{k+2}) + s \cdot C'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^{m+1}_{k+2})\\
&\quad + t \cdot B'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^m_{k+2}) + c'_3(\tilde{d}_{k+1}) \cdot G'_k(\tilde{d}^m_{k+2}) && (91)\\
C'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2}) &= l \cdot C'_{k+1}(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2}) + l \cdot C'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m-1}_{k+2}) && (n \geq 2)\\
&\quad + s \cdot C'_{k+1}(\tilde{d}^{n+1}_{k+1} * \tilde{d}^m_{k+2}) + s \cdot C'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^{m+1}_{k+2})\\
&\quad + t \cdot B'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2}) + c_3(\tilde{d}_{k+1}) \cdot G'_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2}) && (92)
\end{aligned}$$

$$\begin{aligned}
D'_k(\lambda) &= t \cdot E'_k(\lambda) + s_5(d_{k+1}) \cdot F'_k(\lambda) && (93)\\
D'_k(\tilde{d}^n_{k+1}) &= l \cdot D'_k(\tilde{d}^{n-1}_{k+1}) + s \cdot D'_k(\tilde{d}^{n+1}_{k+1}) && (n \geq 1)\\
&\quad + t \cdot E'_k(\tilde{d}^n_{k+1}) + s_5(d_{k+1}) \cdot F'_k(\tilde{d}^n_{k+1}) && (94)
\end{aligned}$$

$$\begin{aligned}
E'_k(\lambda) &= c_2(\tilde{d}_{k+1}) \cdot D'_k(\tilde{d}_{k+1}) + s_5(d_{k+1}) \cdot H'_k(\lambda) && (95)\\
E'_k(\tilde{d}^n_{k+1}) &= l \cdot E'_k(\tilde{d}^{n-1}_{k+1}) + s \cdot E'_k(\tilde{d}^{n+1}_{k+1}) && (n \geq 1)\\
&\quad + c_2(\tilde{d}_{k+1}) \cdot D'_k(\tilde{d}^{n+1}_{k+1}) + s_5(d_{k+1}) \cdot H'_k(\tilde{d}^n_{k+1}) && (96)
\end{aligned}$$

$$\begin{aligned}
F'_k(\lambda) &= t \cdot H'_k(\lambda) + c_4(\overline{k}) \cdot A'_{k+1}(\lambda) && (97)\\
F'_k(\tilde{d}^n_{k+1}) &= l \cdot F'_k(\tilde{d}^{n-1}_{k+1}) + s \cdot F'_k(\tilde{d}^{n+1}_{k+1}) && (n \geq 1)\\
&\quad + t \cdot H'_k(\tilde{d}^n_{k+1}) + c_4(\overline{k}) \cdot A'_{k+1}(\tilde{d}^n_{k+1}) && (98)
\end{aligned}$$

$$\begin{aligned}
G'_k(\lambda) &= t \cdot J'_k(\lambda) + c_4(\overline{k}) \cdot B'_{k+1}(\lambda) && (99)\\
G'_k(\tilde{d}^m_{k+2}) &= l \cdot G'_k(\tilde{d}^{m-1}_{k+2}) + s \cdot G'_k(\tilde{d}^{m+1}_{k+2})\\
&\quad + t \cdot J'_k(\tilde{d}^m_{k+2}) + c_4(\overline{k}) \cdot B'_{k+1}(\tilde{d}^m_{k+2}) && (100)
\end{aligned}$$

$$\begin{aligned}
G'_k(\tilde{d}_{k+1}) &= l' \cdot G'_k(\lambda) + s \cdot G'_k(\tilde{d}^2_{k+1}) \\
&\quad + t \cdot J'_k(\tilde{d}_{k+1}) + c_4(\overline{k}) \cdot B'_{k+1}(\tilde{d}_{k+1}) && (101)\\
G'_k(\tilde{d}^n_{k+1}) &= l \cdot G'_k(\tilde{d}^{n-1}_{k+1}) + s \cdot G'_k(\tilde{d}^{n+1}_{k+1}) && (n \geq 2)\\
&\quad + t \cdot J'_k(\tilde{d}^n_{k+1}) + c_4(\overline{k}) \cdot B'_{k+1}(\tilde{d}^n_{k+1}) && (102)\\
G'_k(\tilde{d}_{k+1} * \tilde{d}^m_{k+2}) &= l' \cdot G'_k(\tilde{d}^m_{k+2}) + l \cdot G'_k(\tilde{d}_{k+1} * \tilde{d}^{m-1}_{k+2}) \\
&\quad + s \cdot G'_k(\tilde{d}^2_{k+1} * \tilde{d}^m_{k+2}) + s \cdot G'_k(\tilde{d}_{k+1} * \tilde{d}^{m+1}_{k+2}) \\
&\quad + t \cdot J'_k(\tilde{d}_{k+1} * \tilde{d}^m_{k+2}) + c_4(\overline{k}) \cdot B'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^m_{k+2}) && (103)\\
G'_k(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2}) &= l \cdot G'_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2}) + l \cdot G'_k(\tilde{d}^n_{k+1} * \tilde{d}^{m-1}_{k+2}) && (n \geq 2)\\
&\quad + s \cdot G'_k(\tilde{d}^{n+1}_{k+1} * \tilde{d}^m_{k+2}) + s \cdot G'_k(\tilde{d}^n_{k+1} * \tilde{d}^{m+1}_{k+2}) \\
&\quad + t \cdot J'_k(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2}) + c_4(\overline{k}) \cdot B'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2}) && (104)
\end{aligned}$$

$$\begin{aligned}
H'_k(\lambda) &= c_2(\tilde{d}_{k+1}) \cdot F'_k(\tilde{d}_{k+1}) && (105)\\
H'_k(\tilde{d}^n_{k+1}) &= l \cdot H'_k(\tilde{d}^{n-1}_{k+1}) + s \cdot H'_k(\tilde{d}^{n+1}_{k+1}) && (n \geq 1)\\
&\quad + c_2(\tilde{d}_{k+1}) \cdot F'_k(\tilde{d}^{n+1}_{k+1}) && (106)
\end{aligned}$$

$$\begin{aligned}
I'_k(\lambda) &= c_1(d_{k+2}) \cdot J'_k(\lambda) && (107)\\
I'_k(\tilde{d}^n_{k+1}) &= l \cdot I'_k(\tilde{d}^{n-1}_{k+1}) + s \cdot I'_k(\tilde{d}^{n+1}_{k+1}) && (n \geq 1)\\
&\quad + c_1(d_{k+2}) \cdot J'_k(\tilde{d}^n_{k+1}) && (108)
\end{aligned}$$

$$\begin{aligned}
J'_k(\lambda) &= c_2(\tilde{d}_{k+2}) \cdot G'_k(c_2(\tilde{d}_{k+2})) && (109)\\
J'_k(\tilde{d}^m_{k+2}) &= l \cdot J'_k(\tilde{d}^{m-1}_{k+2}) + s \cdot J'_k(\tilde{d}^{m+1}_{k+2}) \\
&\quad + c_2(\tilde{d}_{k+2}) \cdot G'_k(\tilde{d}^{m+1}_{k+2}) && (110)\\
J'_k(\tilde{d}_{k+1}) &= l' \cdot J'_k(\lambda) + s \cdot J'_k(\tilde{d}^2_{k+1}) \\
&\quad + c_2(\tilde{d}_{k+2}) \cdot G'_k(\tilde{d}_{k+1} * \tilde{d}_{k+2}) && (111)\\
J'_k(\tilde{d}^n_{k+1}) &= l \cdot J'_k(\tilde{d}^{n-1}_{k+1}) + s \cdot J'_k(\tilde{d}^{n+1}_{k+1}) && (n \geq 2)\\
&\quad + c_2(\tilde{d}_{k+2}) \cdot G'_k(\tilde{d}^n_{k+1} * \tilde{d}_{k+2}) && (112)\\
J'_k(\tilde{d}_{k+1} * \tilde{d}^m_{k+2}) &= l' \cdot J'_k(\tilde{d}^m_{k+2}) + l \cdot J'_k(\tilde{d}_{k+1} * \tilde{d}^{m-1}_{k+2}) \\
&\quad + s \cdot J'_k(\tilde{d}^2_{k+1} * \tilde{d}^m_{k+2}) + s \cdot J'_k(\tilde{d}_{k+1} * \tilde{d}^{m+1}_{k+2}) \\
&\quad + c_2(\tilde{d}_{k+2}) \cdot G'_k(\tilde{d}_{k+1} * \tilde{d}^{m+1}_{k+2}) && (113)\\
J'_k(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2}) &= l \cdot J'_k(\tilde{d}^{n-1}_{k+1} * \tilde{d}^m_{k+2}) + l \cdot J'_k(\tilde{d}^n_{k+1} * \tilde{d}^{m-1}_{k+2}) && (n \geq 2)\\
&\quad + s \cdot J'_k(\tilde{d}^{n+1}_{k+1} * \tilde{d}^m_{k+2}) + s \cdot J'_k(\tilde{d}^n_{k+1} * \tilde{d}^{m+1}_{k+2}) \\
&\quad + c_2(\tilde{d}_{k+2}) \cdot J'_k(\tilde{d}^n_{k+1} * \tilde{d}^{m+1}_{k+2}) && (114)
\end{aligned}$$

Let E' be the guarded recursive specification consisting of the equations (79)–(114) and let E be the guarded recursive specification consisting of the same equations but without the extra acutes on the atomic actions. According to *RDP* there is a solution for the guarded recursive specification E'. We will denote this solution as follows.

$$a'_k(\lambda), a'_{k+1}(\tilde{d}^n_{k+1}), \ldots, j'_k(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2}).$$

We will show that we may use these processes in order to prove the theorem. Therefore, we will introduce a linear unary operator. Let $\{r\}$ be a set of function names and consider the following linear functional specification $E(\{r\})$.

$$\begin{aligned}
E(\{r\}) = &\left\{\, r(l') = l, r\big(c'_3(x)\big) = c_3(x) : x \in E \,\right\} \\
&\cup \left\{\, r(a) = a : a \in A \setminus \{l', c'_3(x) : x \in E\} \,\right\} \\
&\cup \left\{\, r(a \cdot x) = r(a) \cdot r(x) : a \in A \,\right\}.
\end{aligned}$$

In accordance with *ODP* there is a solution for this system, say ϱ. It will be clear that

$$\varrho\big(a'_k(\lambda)\big), \varrho\big(a'_{k+1}(\tilde{d}^n_{k+1})\big), \ldots, \varrho\big(j'_k(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big)$$

is a solution for the guarded recursive specification E without accented atomic actions. But it is also evident that

$$\partial_H\big(A_k(\lambda)\big), \partial_H\big(A_{k+1}(\tilde{d}^n_{k+1})\big), \ldots, \partial_H\big(J_k(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big)$$

is a solution for the guarded recursive specification E. So with the aid of *RSP* we conclude that these solutions are equal and we find, in particular,

$$\varrho\big(a'_0(\lambda)\big) = \partial_H\big(A_0(\lambda)\big). \tag{115}$$

Let $I' = I \cup \{l\} \cup \{c'_3(x) : x \in E\}$ be a subset of the set of atomic actions A. We will prove that $\tau_{I'} = \tau_I \circ \varrho$. Let $\{t\}$ be a functon name and let $E(\{t\})$ be the defining linear functional specification for $\tau_{I'}$.

$$\begin{aligned} E(\{t\}) = &\left\{\, t(a) = a : a \in A \setminus I' \,\right\} \\ &\cup \left\{\, t(a) = \tau : a \in I' \,\right\} \\ &\cup \left\{\, t(a \cdot x) = t(a) \cdot t(x) : a \in A \,\right\}. \end{aligned}$$

It will be clear that $\tau_{I'}$ solves this system. It is not hard to see that $\tau_I \circ \varrho$ is also a solution for the linear functional specification $E(\{t\})$. So in accord with *OSP* we find that these solutions must be equal and $\tau_{I'} = \tau_I \circ \varrho$. With this in mind we can deduce the following relation between a'_0 and A_0 using equation (115).

$$\begin{aligned} \tau_{I'}\big(a'_0(\lambda)\big) &= \tau_I \circ \varrho\big(a'_0(\lambda)\big) \\ &= \tau_I \circ \partial_H\big(A_0(\lambda)\big). \end{aligned}$$

So in order to prove theorem (4.16) it will be sufficient to deduce that the following equation is valid.

$$\tau_{I'}\big(a'_0(\lambda)\big) = \tau \cdot \prod_{i=1}^{\infty} s_5(d_i). \tag{116}$$

The last part of this proof will consist of the verification of equation (116). So for convenience sake we will stipulate the structure of this last part. We will make an assumption on a *DE*-trace for a fixed $p \geq 1$. Then we will state a conclusion on an *FH*-trace that can be derived from this assumption. Then the actual derivation of this will follow. It will be easy to see that the assumption on this *DE*-trace is valid for $p = 1$. With the aid of this we can set off a chain reaction and in the end we will find that both the assumption on the *DE*-trace and the conclusion on the *FH*-trace are true statements by themselves. With this knowledge we can make a small calcluation on a single *BC*-trace. Then a straightforward deduction in which we will use *GIP* will end the proof of (116).

Let $J = \{s_5(d) : d \in D\}$ be a subset of the set of atomic actions A. Let $\beta_1 = \beta(I', J)$. Let $p \geq 1$ be arbitrary but fixed. Suppose that the following holds for all $k \geq 0$ and $n \geq 1$.

$$\pi^p_J \circ \beta_1\big(d'_k(\lambda)\big) = t \cdot \pi^p_J \circ \beta_1\big(e'_k(\lambda)\big) + \prod_{i=1}^{p} s_5(d_{k+i}) \tag{117}$$

$$\begin{aligned} \pi^p_J \circ \beta_1\big(d'_k(\tilde{d}^n_{k+1})\big) = &\ l \cdot \pi^p_J \circ \beta_1\big(d'_k(\tilde{d}^{n-1}_{k+1})\big) + s \cdot \pi^p_J \circ \beta_1\big(d'_k(\tilde{d}^{n+1}_{k+1})\big) \\ &+ t \cdot \pi^p_J \circ \beta_1\big(e'_k(\tilde{d}^n_{k+1})\big) + \prod_{i=1}^{p} s_5(d_{k+i}) \end{aligned} \tag{118}$$

$$\pi^p_J \circ \beta_1\big(e'_k(\lambda)\big) = c_2(\tilde{d}_{k+1}) \cdot \pi^p_J \circ \beta_1\big(e'_k(\tilde{d}_{k+1})\big) + \prod_{i=1}^{p} s_5(d_{k+i}) \tag{119}$$

$$\begin{aligned} \pi^p_J \circ \beta_1\big(e'_k(\tilde{d}^n_{k+1})\big) = &\ l \cdot \pi^p_J \circ \beta_1\big(e'_k(\tilde{d}^{n-1}_{k+1})\big) + s \cdot \pi^p_J \circ \beta_1\big(e'_k(\tilde{d}^{n+1}_{k+1})\big) \\ &+ c_2(\tilde{d}_{k+1}) \cdot \pi^p_J \circ \beta_1\big(d'_k(\tilde{d}^{n+1}_{k+1})\big) + \prod_{i=1}^{p} s_5(d_{k+i}). \end{aligned} \tag{120}$$

Then the following can be derived from these four equations for all $k, n \geq 0$:

$$\left.\begin{array}{l} \pi_J^p \circ \tau_{I'}\big(f_k'(\tilde{d}_{k+1}^n)\big) \\ \pi_J^p \circ \tau_{I'}\big(h_k'(\tilde{d}_{k+1}^n)\big) \end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{121}$$

In order to derive this we will first verify a similar result from the assumptions on double BC-traces and the JG-traces. In the subsequent display we will state what we are going to deduce from the assumptions for all $k, u, m \geq 0$.

$$\left.\begin{array}{l} \pi_J^p \circ \tau_{I'}\big(b_{k+1}'(\tilde{d}_{k+1}^u * \tilde{d}_{k+2}^m)\big) \\ \pi_J^p \circ \tau_{I'}\big(c_{k+1}'(\tilde{d}_{k+1}^u * \tilde{d}_{k+2}^{m+1})\big) \\ \pi_J^p \circ \tau_{I'}\big(g_k'(\tilde{d}_{k+1}^u * \tilde{d}_{k+2}^{m+1})\big) \\ \pi_J^p \circ \tau_{I'}\big(j_k'(\tilde{d}_{k+1}^u * \tilde{d}_{k+2}^m)\big) \end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{122}$$

In the former theorems we verified this with induction to u. In fact, we proved it with induction to the number of old datum elements available. This number could not be increased once we entered a double BC-trace or a JG-trace. So it was convenient to use this induction on the floor height of the traces. If we had assumed that the queue could only duplicate "new" datum elements, we could still use the floor height induction argument. For, we would not be able to reach a higher floor once we entered a double BC-trace or a JG-trace. And the caculations in the previous theorems could have been reiterated. Now we are in a different situation: as soon as we enter a trace that contains just one old datum element we can reach every higher situated horizontal trace of the dense block, q.v. figure 10. However, the proof for the case $u = 0$ will be the same as in the former theorems since if $u = 0$ there are no old datum elements in the queue. We will use this result to calculate the general case $u \geq 0$. So let $u = 0$ and fix an arbitrary $k \geq 0$. Consider the following guarded recursive specification E_1.

$$\begin{aligned} E_1 = \big\{\, X_0^1 &= t \cdot Y_0^1 + Z_1 \\ Y_0^1 &= c_2(\tilde{d}_{k+2}) \cdot X_1^1 + Z_1 \\ X_n^1 &= l \cdot X_{n-1}^1 + s \cdot X_{n+1}^1 + t \cdot Y_n^1 + Z_1 \\ Y_n^1 &= l \cdot Y_{n-1}^1 + s \cdot Y_{n+1}^1 + c_2(\tilde{d}_{k+2}) \cdot X_{n+1}^1 + Z_1 \\ Z_1 &= \prod_{i=1}^{p} s_5(d_{k+1+i}) \;\big|\; n \geq 1 \,\big\}. \end{aligned}$$

Let $C_1 = \{X_n^1, Y_n^1 : n \geq 0\}$. Then C_1 is a conservative cluster of E_1 in I'. Since the set of exits $U(C_1) = \{Z_1\}$ is finite, we may apply $CFAR^\infty$ and we find for all $n \geq 0$:

$$\tau_{I'}(X_n^1) = \tau_{I'}(Y_n^1) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Because of equations (117)–(120) we see that if we put for all $n \geq 0$

$$\begin{aligned} X_n^1 &= \pi_J^p \circ \beta_1\big(d_{k+1}'(\tilde{d}_{k+2}^n)\big) \\ Y_n^1 &= \pi_J^p \circ \beta_1\big(e_{k+1}'(\tilde{d}_{k+2}^n)\big), \end{aligned}$$

this will solve the guarded recursive specification E_1. Now consider the following calculation.

$$\begin{aligned} \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) &= \tau_{I'}(X_n^1) \\ &= \tau_{I'} \circ \pi_J^p \circ \beta_1\big(d_{k+1}'(\tilde{d}_{k+2}^n)\big) \\ &= \pi_J^p \circ \tau_{I'} \circ \beta_1\big(d_{k+1}'(\tilde{d}_{k+2}^n)\big) && \text{because of (3.12)} \\ &= \pi_J^p \circ \tau_{I'}\big(d_{k+1}'(\tilde{d}_{k+2}^n)\big) && \text{see lemma (4.6)} \end{aligned}$$

Such a calculation can be made for e'_{k+1}, too. We find thus for all $n \geq 0$

$$\left.\begin{array}{l} \pi_J^p \circ \tau_{I'}\big(d'_{k+1}(\tilde{d}_{k+2}^n)\big) \\ \pi_J^p \circ \tau_{I'}\big(e'_{k+1}(\tilde{d}_{k+2}^n)\big) \end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{123}$$

Now consider the following guarded recursive specification E_2.

$$\begin{aligned} E_2 = \{\ & X_0^2 = c_2(\tilde{d}_{k+2}) \cdot Y_1^2 \\ & Y_0^2 = t \cdot X_0^2 \\ & X_n^2 = l \cdot X_{n-1}^2 + s \cdot X_{n+1}^2 + c_2(\tilde{d}_{k+2}) \cdot Y_{n+1}^2 + Z_2 \\ & Y_n^2 = l \cdot Y_{n-1}^2 + s \cdot Y_{n+1}^2 + t \cdot X_n^2 + Z_2 \\ & Z_2 = c_3(\tilde{d}_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \mid n \geq 1 \}. \end{aligned}$$

Let $C_2 = \{X_n^2, Y_n^2 : n \geq 0\}$. Then C_2 is a conservative cluster of E_2 in I'. Since the set of exits $U(C_2) = \{Z_2\}$ is finite, we may apply $CFAR^\infty$ and we find for all $n \geq 0$:

$$\tau_{I'}(X_n^2) = \tau_{I'}(Y_n^2) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Let $\beta_2 = \beta\big(I', \{c_3(\tilde{d}_{k+2})\}\big)$. Hereinafter, we will show that if we put for all $n \geq 0$

$$\begin{aligned} X_n^2 &= \pi_J^p \circ \beta_2\big(b'_{k+1}(\tilde{d}_{k+2}^n)\big) \\ Y_n^2 &= \pi_J^p \circ \beta_2\big(c'_{k+1}(\tilde{d}_{k+2}^n)\big), \end{aligned} \tag{124}$$

this will solve the guarded recursive specification E_2. For $n = 0$ it is trivial to see that, with the aid of equations (81) and (87), this is true for the first two equations of E_2. Now let $n \geq 1$. We will verify the third equation of E_2.

$$\begin{aligned} \pi_J^p \circ \beta_2\big(b'_{k+1}(\tilde{d}_{k+2}^n)\big) &\overset{(82)}{=} l \cdot \pi_J^p \circ \beta_2\big(b'_{k+1}(\tilde{d}_{k+2}^{n-1})\big) \\ &\quad + s \cdot \pi_J^p \circ \beta_2\big(b'_{k+1}(\tilde{d}_{k+2}^{n+1})\big) \\ &\quad + c_2(\tilde{d}_{k+2}) \cdot \pi_J^p \circ \beta_2\big(c'_{k+1}(\tilde{d}_{k+2}^{n+1})\big) \\ &\quad + c_3(\tilde{d}_{k+2}) \cdot \pi_J^p \circ \tau_{I'}\big(e'_{k+1}(\tilde{d}_{k+2}^{n-1})\big) \\ &\overset{(123)}{=} l \cdot \pi_J^p \circ \beta_2\big(b'_{k+1}(\tilde{d}_{k+2}^{n-1})\big) \\ &\quad + s \cdot \pi_J^p \circ \beta_2\big(b'_{k+1}(\tilde{d}_{k+2}^{n+1})\big) \\ &\quad + c_2(\tilde{d}_{k+2}) \cdot \pi_J^p \circ \beta_2\big(c'_{k+1}(\tilde{d}_{k+2}^{n+1})\big) \\ &\quad + c_3(\tilde{d}_{k+2}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

So we see that this solves the equation in E_2 for X_n^2. A similar calculation will yield the same result for the fourth equation of E_2. So the claim that (124) is a solution for E_2 is valid. We find

$$\begin{aligned} \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) &= \tau_{I'}(X_n^2) \\ &= \tau_{I'} \circ \pi_J^p \circ \beta_2\big(b'_{k+1}(\tilde{d}_{k+2}^n)\big) \\ &= \pi_J^p \circ \tau_{I'} \circ \beta_2\big(b'_{k+1}(\tilde{d}_{k+2}^n)\big) && \text{because of (3.12)} \\ &= \pi_J^p \circ \tau_{I'}\big(b'_{k+1}(\tilde{d}_{k+2}^n)\big). && \text{see lemma (4.6)} \end{aligned}$$

Such a calculation can be made for c'_{k+1}, too. Summarizing we have for all $n \geq 0$

$$\left.\begin{array}{l}\pi_J^p \circ \tau_{I'}\big(b'_{k+1}(\tilde{d}^n_{k+2})\big)\\ \pi_J^p \circ \tau_{I'}\big(c'_{k+1}(\tilde{d}^n_{k+2})\big)\end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{125}$$

This means that the first two equations of (122) are proved for $u = 0$ and our fixed $k \geq 0$. Consider the next guarded recursive specification E_3.

$$\begin{aligned} E_3 = \{\ X_0^3 &= t \cdot Y_0^3 + Z_3 \\ Y_0^3 &= c_2(\tilde{d}_{k+2}) \cdot X_1^3 \\ X_m^3 &= l \cdot X_{m-1}^3 + s \cdot X_{m+1}^3 + t \cdot Y_m^3 + Z_3 \\ Y_m^3 &= l \cdot Y_{m-1}^3 + s \cdot Y_{m+1}^3 + c_2(\tilde{d}_{k+2}) \cdot X_{m+1}^3 \\ Z_3 &= c_4(\overline{k}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \ \big|\ n \geq 1\ \}. \end{aligned}$$

Let $C_3 = \{X_m^3, Y_m^3 : m \geq 0\}$. Then C_3 is a conservative cluster of E_3 in I'. Since the set of exits $U(C_3) = \{Z_3\}$ is finite, we may apply $CFAR^\infty$ and we find for all $n \geq 0$:

$$\tau_{I'}(X_m^3) = \tau_{I'}(Y_m^3) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Let $\beta_3 = \beta\big(I', \{c_4(\overline{k})\}\big)$. It will not be difficult to see that if we put for all $m \geq 0$

$$\begin{aligned} X_m^3 &= \pi_J^p \circ \beta_3\big(g'_k(\tilde{d}^m_{k+2})\big) \\ Y_m^3 &= \pi_J^p \circ \beta_3\big(j'_k(\tilde{d}^m_{k+2})\big), \end{aligned} \tag{126}$$

this will solve the system E_3. We will only verify the first equation for E_3. Consider thereunto the following display.

$$\begin{aligned} \pi_J^p \circ \beta_3\big(g'_k(\lambda)\big) &\overset{(99)}{=} t \cdot \pi_J^p \circ \beta_3\big(j'_k(\lambda)\big) + c_4(\overline{k}) \cdot \pi_J^p \circ \tau_{I'}\big(b'_{k+1}(\lambda)\big) \\ &\overset{(125)}{=} t \cdot \pi_J^p \circ \beta_3\big(j'_k(\lambda)\big) + c_4(\overline{k}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \end{aligned}$$

The other equations can be handled in the same way while using equation (125). Thus, we find that (126) solves the guarded recursive specification E_3. When we use the fact that the generalized projection operator and the abstraction operator commute and the fact that the abstraction operator absorbs the selective abstraction operator β_3 we will find with a calculation that is similar to the ones we did for E_1 and E_2 that for all $m \geq 0$

$$\left.\begin{array}{l}\pi_J^p \circ \tau_{I'}\big(g'_k(\tilde{d}^m_{k+2})\big)\\ \pi_J^p \circ \tau_{I'}\big(j'_k(\tilde{d}^m_{k+2})\big)\end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Now we find that the second two equations of (122) are also correct so we see that the case $u = 0$ is proved for (122), since $k \geq 0$ was arbitrarily chosen. We will use this result to prove equation (122) for all $u \geq 0$. We can think of the dense block as a very big (conservative) cluster. The only way out for this cluster is by removing all the old data from the queue, that is, the case $u = 0$. Precisely the atomic actions that result in this situation are acuted. So if we make a β that transforms into the abstraction operator after passing the acuted actions we can use the result for $u = 0$. To achieve this consider the subsequent equations. The guarded recursive specification E_4 consists of these equations for all $n \geq 2$ and $m \geq 1$. Let $k \geq 0$ be fixed.

$$\begin{aligned}
X^4_{1,0} &= l' \cdot Z_4 + s \cdot X^4_{2,0} + c_2(\tilde{d}_{k+2}) \cdot Y^4_{1,1} + c'_3(\tilde{d}_{k+1}) \cdot Z_4 \\
X^4_{n,0} &= l \cdot X^4_{n-1,0} + s \cdot X^4_{n+1,0} + c_2(\tilde{d}_{k+2}) \cdot Y^4_{n,1} + c_3(\tilde{d}_{k+1}) \cdot T^4_{n-1,0} \\
X^4_{1,m} &= l' \cdot Z_4 + l \cdot X^4_{1,m-1} + s \cdot X^4_{2,m} + s \cdot X^4_{1,m+1} \\
&\quad + c_2(\tilde{d}_{k+2}) \cdot Y^4_{1,m+1} + c'_3(\tilde{d}_{k+1}) \cdot Z_4 \\
X^4_{n,m} &= l \cdot X^4_{n-1,m} + l \cdot X^4_{n,m-1} + s \cdot X^4_{n+1,m} + s \cdot X^4_{n,m+1} \\
&\quad + c_2(\tilde{d}_{k+2}) \cdot Y^4_{n,m+1} + c_3(\tilde{d}_{k+1}) \cdot T^4_{n-1,m} \\
Y^4_{1,0} &= l' \cdot Z_4 + s \cdot Y^4_{2,0} + t \cdot X^4_{1,0} + c'_3(\tilde{d}_{k+1}) \cdot Z_4 \\
Y^4_{n,0} &= l \cdot Y^4_{n-1,0} + s \cdot Y^4_{n+1,0} + t \cdot X^4_{n,0} + c_3(\tilde{d}_{k+1}) \cdot Z^4_{n-1,0} \\
Y^4_{1,m} &= l' \cdot Z_4 + l \cdot Y^4_{1,m-1} + s \cdot Y^4_{2,m} + s \cdot Y^4_{1,m+1} \\
&\quad + t \cdot X^4_{1,m} + c'_3(\tilde{d}_{k+1}) \cdot Z_4 \\
Y^4_{n,m} &= l \cdot Y^4_{n-1,m} + l \cdot Y^4_{n,m-1} + s \cdot Y^4_{n+1,m} + s \cdot Y^4_{n,m+1} \\
&\quad + t \cdot X^4_{n,m} + c_3(\tilde{d}_{k+1}) \cdot Z^4_{n-1,m} \\
Z^4_{1,0} &= l' \cdot Z_4 + s \cdot Z^4_{2,0} + t \cdot T^4_{1,0} + c_4(\overline{k}) \cdot X^4_{1,0} \\
Z^4_{n,0} &= l \cdot Z^4_{n-1,0} + s \cdot Z^4_{n+1,0} + t \cdot T^4_{n,0} + c_4(\overline{k}) \cdot X^4_{n,0} \\
Z^4_{1,m} &= l' \cdot Z_4 + l \cdot Z^4_{1,m-1} + s \cdot Z^4_{2,m} + s \cdot Z^4_{1,m+1} \\
&\quad + t \cdot T^4_{1,m} + c_4(\overline{k}) \cdot X^4_{1,m} \\
Z^4_{n,m} &= l \cdot Z^4_{n-1,m} + l \cdot Z^4_{n,m-1} + s \cdot Z^4_{n+1,m} + s \cdot Z^4_{n,m+1} \\
&\quad + t \cdot T^4_{n,m} + c_4(\overline{k}) \cdot X^4_{n,m} \\
T^4_{1,0} &= l' \cdot Z_4 + s \cdot T^4_{2,0} + c_2(\tilde{d}_{k+2}) \cdot Z^4_{1,1} \\
T^4_{n,0} &= l \cdot T^4_{n-1,0} + s \cdot T^4_{n+1,0} + c_2(\tilde{d}_{k+2}) \cdot Z^4_{n,1} \\
T^4_{1,m} &= l' \cdot Z_4 + l \cdot T^4_{1,m-1} + s \cdot T^4_{2,m} + s \cdot T^4_{1,m+1} \\
&\quad + c_2(\tilde{d}_{k+2}) \cdot Z^4_{1,m+1} \\
T^4_{n,m} &= l \cdot T^4_{n-1,m} + l \cdot T^4_{n,m-1} + s \cdot T^4_{n+1,m} + s \cdot T^4_{n,m+1} \\
&\quad + c_2(\tilde{d}_{k+2}) \cdot Z^4_{n,m+1} \\
Z_4 &= \prod_{i=1}^{p} s_5(d_{k+1+i})
\end{aligned}$$

Let $C_4 = \{X^4_{n,m}, Y^4_{n,m}, Z^4_{n,m}, T^4_{n,m} : n \geq 1, m \geq 0\}$. The set of exits is $U(C_4) = \{l' \cdot Z_4, c'_3(\tilde{d}_{k+1}) \cdot Z_4\}$. It is not hard to see that C_4 is a conservative cluster of E_4 in I'; so since the set of exits is finite we may apply $CFAR^\infty$ and we find for all $n \geq 1$ and $m \geq 0$

$$\tau_{I'}(X^4_{n,m}) = \tau_{I'}(Y^4_{n,m}) = \tau_{I'}(Z^4_{n,m}) = \tau_{I'}(T^4_{n,m}) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{127}$$

Let $\beta_4 = \beta\big(I', \{l', c_3(\tilde{d}_{k+1})\}\big)$. It will be not that hard to see that, if we put for all $n \geq 1$ and $m \geq 0$

$$\begin{aligned}
X^4_{n,m} &= \pi^p_J \circ \beta_4\big(b'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big) \\
Y^4_{n,m} &= \pi^p_J \circ \beta_4\big(c'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big) \\
Z^4_{n,m} &= \pi^p_J \circ \beta_4\big(g'_k(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big) \\
T^4_{n,m} &= \pi^p_J \circ \beta_4\big(j'_k(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big)
\end{aligned}$$

this will solve the guarded recursive specification E_4. To show the usage of the acuted atomic actions we will prove the first four equations of the guarded recursive specification E_4.

$$\begin{aligned} X^4_{1,0} &= \pi^p_J \circ \beta_4\big(b'_{k+1}(\tilde{d}_{k+1})\big) \\ &= l' \cdot \pi^p_J \circ \tau_{I'}\big(b'_{k+1}(\lambda)\big) + s \cdot \pi^p_J \circ \beta_4\big(b'_{k+1}(\tilde{d}^2_{k+1})\big) && \text{because of (83)} \\ &\quad + c_2(\tilde{d}_{k+2}) \cdot \pi^p_J \circ \beta_4\big(c'_{k+1}(\tilde{d}_{k+1} * \tilde{d}_{k+2})\big) \\ &\quad + c'_3(\tilde{d}_{k+1}) \cdot \pi^p_J \circ \tau_{I'}\big(j'_k(\lambda)\big) \\ &= l' \cdot Z_4 + s \cdot X^4_{2,0} + c_2(\tilde{d}_{k+2}) \cdot Y^4_{1,1} + c'_3(\tilde{d}_{k+1}) \cdot Z_4. && \text{use (122) for } u = 0 \end{aligned}$$

The second equation. Let $n \geq 2$. In equation (84) there are no acuted atomic actions so β_4 will not transform into an abstraction operator.

$$\begin{aligned} X^4_{n,0} &= \pi^p_J \circ \beta_4\big(b'_{k+1}(\tilde{d}^n_{k+1})\big) \\ &= l \cdot X^4_{n-1,0} + s \cdot X^4_{n+1,0} + c_2(\tilde{d}_{k+2}) \cdot Y^4_{n,1} + c_3(\tilde{d}_{k+1}) \cdot T^4_{n-1,0}. \end{aligned}$$

The third equation.

$$\begin{aligned} X^4_{1,m} &= \pi^p_J \circ \beta_4\big(b'_{k+1}(\tilde{d}_{k+1} * \tilde{d}^m_{k+2})\big) \\ &= l' \cdot \pi^p_J \circ \tau_{I'}\big(b'_{k+1}(\tilde{d}^m_{k+2})\big) && \text{because of (85)} \\ &\quad + l \cdot X^4_{1,m-1} + s \cdot X^4_{2,m} + s \cdot X^4_{1,m+1} \\ &\quad + c_2(\tilde{d}_{k+2}) \cdot Y^4_{1,m+1} + c'_3(\tilde{d}_{k+1}) \cdot \pi^p_J \circ \tau_{I'}\big(j'_k(\tilde{d}^m_{k+2})\big) \\ &= l' \cdot Z_4 + l \cdot X^4_{1,m-1} + s \cdot X^4_{2,m} + s \cdot X^4_{1,m+1} && \text{use (122) for } u = 0 \text{ twice} \\ &\quad + c_2(\tilde{d}_{k+2}) \cdot Y^4_{1,m+1} + c'_3(\tilde{d}_{k+1}) \cdot Z_4 \end{aligned}$$

The fourth equation. In equation (86) there are no accented atomic actions so β_4 will not do anything. Just as with the second equation the verification of the fourth equation is trivial. The other equations are verified in exactly the same way as these first four. Recall that we are proving (122). We will prove that the first equation of (122) holds.

$$\begin{aligned} \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) &= \tau_{I'}(X^4_{n,m}) && \text{with (127)} \\ &= \tau_{I'} \circ \pi^p_J \circ \beta_4\big(b'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big) \\ &= \pi^p_J \circ \tau_{I'} \circ \beta_4\big(b'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big) && \text{because of (3.12)} \\ &= \pi^p_J \circ \tau_{I'}\big(b'_{k+1}(\tilde{d}^n_{k+1} * \tilde{d}^m_{k+2})\big) && \text{see lemma (4.6)} \end{aligned}$$

Since this deduction is valid for all $n \geq 1$ and $m \geq 0$ we see that the first equation of (122) is correct. The other three are verified analogously. We have also seen that (122) is correct for $u = 0$ so this will end the proof of (122).

Now we will verify that from equations (117)–(120) we can derive (121). Choose a fixed $k \geq 0$. First, we will prove for all $n \geq 0$ for the *I*-nodes:

$$\pi^p_J \circ \tau_{I'}\big(i'_k(\tilde{d}^n_{k+1})\big) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{128}$$

Consider thereunto the following guarded recursive specification E_5.

$$\begin{aligned} E_5 = \Big\{\, & X^5_0 = c_1(d_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ & X^5_n = l \cdot X^5_{n-1} + s \cdot X^5_{n+1} + X^5_0 \;\big|\; n \geq 1 \,\Big\}. \end{aligned}$$

Let $C_5 = \{X_n^5 : n \geq 1\}$ and $U(C_5) = \{X_0^5\}$. We see that C_5 is a conservative cluster for E_5 in I' hence, we may apply *CFAR*$^\infty$ and we find for all $n \geq 1$

$$\tau_{I'}(X_n^5) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) = \tau_{I'}(X_0^5).$$

Let $\beta_5 = \beta\bigl(I', \{c_1(d_{k+2})\}\bigr)$. We will show that if we put for all $n \geq 0$

$$X_n^5 = \pi_J^p \circ \beta_5\bigl(i'_k(\tilde{d}_{k+1}^n)\bigr)$$

this will solve the system E_5. We begin with the first equation.

$$\begin{aligned} X_0^5 &= \pi_J^p \circ \beta_5\bigl(i'_k(\lambda)\bigr) \\ &= c_1(d_{k+2}) \cdot \pi_J^p \circ \tau_{I'}\bigl(j'_k(\lambda)\bigr) && \text{see (107)} \\ &= c_1(d_{k+2}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). && \text{use (122)} \end{aligned}$$

Now let $n \geq 1$.

$$\begin{aligned} X_n^5 &= \pi_J^p \circ \beta_5\bigl(i'_k(\tilde{d}_{k+1}^n)\bigr) \\ &= l \cdot X_{n-1}^5 + s \cdot X_{n+1}^5 + c_1(d_{k+2}) \cdot \pi_J^p \circ \tau_{I'}\bigl(j'_k(\tilde{d}_{k+1}^n)\bigr) && \text{because of (108)} \\ &= l \cdot X_{n-1}^5 + s \cdot X_{n+1}^5 + c_1(d_{k+2}) \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) && \text{use (122)} \\ &= l \cdot X_{n-1}^5 + s \cdot X_{n+1}^5 + X_0^5. \end{aligned}$$

Using theorem (3.12) and lemma (4.6) we will find that equation (128) is correct.

With the use of (128) we are going to verify that a similar result holds for the A-nodes. For all $n \geq 1$ the following is valid

$$\pi_J^p \circ \tau_{I'}\bigl(a'_{k+1}(\tilde{d}_{k+1}^n)\bigr) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}). \tag{129}$$

Consider the subsequent guarded recursive specification E_6.

$$\begin{aligned} E_6 = \{\ X_0^6 &= c_1(d_{k+2}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \\ X_n^6 &= l \cdot X_{n-1}^6 + s \cdot X_{n+1}^6 + X_0^6 + Z_6 \\ Z_6 &= c_3(\tilde{d}_{k+1}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \ \big|\ n \geq 1\ \}. \end{aligned}$$

Let $C_6 = \{X_n^6 : n \geq 1\}$ and $U(C_6) = \{X_0^6, Z_6\}$. It is easy to see that C_6 is a conservative cluster for E_6 in I' so in accord with *CFAR*$^\infty$ we find for all $n \geq 1$

$$\tau_{I'}(X_n^6) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) = \tau_{I'}(X_0^6).$$

Let $\beta_6 = \beta\bigl(I', \{c_1(d_{k+2}), c_3(\tilde{d}_{k+1})\}\bigr)$. It is not difficult to see that if we put for all $n \geq 0$

$$X_n^6 = \pi_J^p \circ \beta_6\bigl(a'_{k+1}(\tilde{d}_{k+1}^n)\bigr)$$

this will solve the specification E_6. We will show this for the second equation of E_6.

$$\begin{aligned}
X_n^6 &= \pi_J^p \circ \beta_6\big(a'_{k+1}(\tilde{d}_{k+1}^n)\big) \\
&= l \cdot X_{n-1}^6 + s \cdot X_{n+1}^6 + c_1(d_{k+2}) \cdot \pi_J^p \circ \tau_{I'}\big(b'_{k+1}(\tilde{d}_{k+1}^n)\big) \\
&\qquad + c_3(\tilde{d}_{k+1}) \cdot \pi_J^p \circ \tau_{I'}\big(i'_k(\tilde{d}_{k+1}^{n-1})\big) && \text{see (80)} \\
&= l \cdot X_{n-1}^6 + s \cdot X_{n+1}^6 + X_0^6 && \text{because of (122)} \\
&\qquad + c_3(\tilde{d}_{k+1}) \cdot \pi_J^p \circ \tau_{I'}\big(i'_k(\tilde{d}_{k+1}^{n-1})\big) \\
&= l \cdot X_{n-1}^6 + s \cdot X_{n+1}^6 + X_0^6 + Z_6. && \text{use (128)}
\end{aligned}$$

And using (3.12) and (4.6) we find that equation (129) is verified.

Now we are in a position to prove that equation (121) is correct. Consider thereunto the following guarded recursive specification E_7.

$$\begin{aligned}
E_7 = \big\{\ X_0^7 &= t \cdot Y_0^7 + Z_7 \\
Y_0^7 &= c_2(\tilde{d}_{k+1}) \cdot X_1^7 \\
X_n^7 &= l \cdot X_{n-1}^7 + s \cdot X_{n+1}^7 + t \cdot Y_n^7 + Z_7 \\
Y_n^7 &= l \cdot Y_{n-1}^7 + s \cdot Y_{n+1}^7 + c_2(\tilde{d}_{k+1}) \cdot X_{n+1}^7 \\
Z_7 &= c_4(\overline{k}) \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}) \ \big|\ n \geq 1 \,\big\}.
\end{aligned}$$

Let $C_7 = \{X_n^7, Y_n^7 : n \geq 0\}$ and $U(C_7) = \{Z_7\}$. We see that C_7 is a conservative cluster for E_7 in I' so we can apply $CFAR^\infty$ and we find for all $n \geq 0$

$$\tau_{I'}(X_n^7) = \tau_{I'}(Y_n^7) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+1+i}).$$

Let $\beta_7 = \beta\big(I', \{c_4(\overline{k})\}\big)$. If we put for all $n \geq 0$

$$\begin{aligned}
X_n^7 &= \pi_J^p \circ \beta_7\big(f'_k(\tilde{d}_{k+1}^n)\big) \\
Y_n^7 &= \pi_J^p \circ \beta_7\big(h'_k(\tilde{d}_{k+1}^n)\big)
\end{aligned}$$

this will solve the guarded recursive specification E_7. We will verify the first equation. The other equations are verified analogously.

$$\begin{aligned}
X_0^7 &= \pi_J^p \circ \beta_7\big(f'_k(\lambda)\big) \\
&= t \cdot Y_0^7 + c_4(\overline{k}) \cdot \pi_J^p \circ \tau_{I'}\big(a'_{k+1}(\lambda)\big) && \text{because of (97)} \\
&= t \cdot Y_0^7 + Z_7. && \text{use (129)}
\end{aligned}$$

Combining these facts while using theorem (3.12) and lemma (4.6) we find that equation (121) is correct under the assumption that equations (117)–(120) hold.

At this point we will prove that without these assumptions equation (121) is also valid. We will show this with induction on p. Let $p = 1$. We will show that the equations (117)–(120) are valid for $p = 1$. Let us first consider equation (117) for $p = 1$. With the aid of equation (93) we see immediately

$$\pi_J^1 \circ \beta_1\big(d'_k(\lambda)\big) = t \cdot \pi_J^1 \circ \beta_1\big(e'_k(\lambda)\big) + s_5(d_{k+1}).$$

This is precisely assumption (117). Now we will look at the second assumption (118). With the aid of equation (94) we see at once

$$\begin{aligned}\pi_J^1\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^n)\big) &= l\cdot\pi_J^1\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^{n-1})\big)+s\cdot\pi_J^1\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^{n+1})\big)\\ &\quad+t\cdot\pi_J^1\circ\beta_1\big(e_k'(\tilde{d}_{k+1}^n)\big)+s_5(d_{k+1}).\end{aligned}$$

This is just equation (118). The remaining two assumptions are verified in the same trivial way. This means that equation (121) is valid for $p=1$, which concludes the basis of our induction. Now suppose that (121) is correct for one $p\geq 1$. Then we will prove it for $p+1$. Thereunto, we will show that the assumptions (117)–(120) are valid for $p+1$. Let us take a look at the first assumption. With the aid of equation (93) we find easily

$$\begin{aligned}\pi_J^{p+1}\circ\beta_1\big(d_k'(\lambda)\big) &= t\cdot\pi_J^{p+1}\circ\beta_1\big(e_k'(\lambda)\big)+s_5(d_{k+1})\cdot\pi_J^p\circ\tau_{I'}\big(f_k'(\lambda)\big)\\ &= t\cdot\pi_J^{p+1}\circ\beta_1\big(e_k'(\lambda)\big)\\ &\quad+s_5(d_{k+1})\cdot\tau\cdot\prod_{i=1}^{p}s_5(d_{k+1+i}) && \text{induction hypothesis}\\ &= t\cdot\pi_J^{p+1}\circ\beta_1\big(e_k'(\lambda)\big)+\prod_{i=1}^{p+1}s_5(d_{k+i}).\end{aligned}$$

This means that the first assumption is satisfied. Below we will show this for the second assumption.

$$\begin{aligned}\pi_J^{p+1}\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^n)\big) &= l\cdot\pi_J^{p+1}\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^{n-1})\big)+s\cdot\pi_J^{p+1}\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^{n+1})\big) && \text{use (94)}\\ &\quad+t\cdot\pi_J^{p+1}\circ\beta_1\big(e_k'(\tilde{d}_{k+1}^n)\big)+s_5(d_{k+1})\cdot\pi_J^p\circ\tau_{I'}\big(f_k'(\tilde{d}_{k+1}^n)\big)\\ &= l\cdot\pi_J^{p+1}\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^{n-1})\big)+s\cdot\pi_J^{p+1}\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^{n+1})\big)\\ &\quad+t\cdot\pi_J^{p+1}\circ\beta_1\big(e_k'(\tilde{d}_{k+1}^n)\big)\\ &\quad+s_5(d_{k+1})\cdot\tau\cdot\prod_{i=1}^{p}s_5(d_{k+1+i}) && \text{induction hypothesis}\\ &= l\cdot\pi_J^{p+1}\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^{n-1})\big)+s\cdot\pi_J^{p+1}\circ\beta_1\big(d_k'(\tilde{d}_{k+1}^{n+1})\big)\\ &\quad+t\cdot\pi_J^{p+1}\circ\beta_1\big(e_k'(\tilde{d}_{k+1}^n)\big)+\prod_{i=1}^{p+1}s_5(d_{k+i}).\end{aligned}$$

The other two assumptions can be treated in exactly the same way. So we find that all the assumptions are valid for $p+1$. But then it follows that equation (121) is proved for $p+1$. This concludes the induction step. Now that we have seen that (121) itself is correct we will use this to show that now also the assumptions themselves are correct. With that result we are going to prove that the following holds for all $n\geq 0$.

$$\left.\begin{array}{l}\pi_J^p\circ\tau_I\big(d_k'(\tilde{d}_{k+1}^n)\big)\\ \pi_J^p\circ\tau_I\big(e_k'(\tilde{d}_{k+1}^n)\big)\end{array}\right\}=\tau\cdot\prod_{i=1}^{p}s_5(d_{k+i}). \tag{130}$$

Consider the subsequent guarded recursive specification E_8.

$$\begin{aligned}E_8=\{\,X_0^8 &= t\cdot Y_0^8+Z_8\\ Y_0^8 &= c_2(\tilde{d}_{k+1})\cdot X_1^8+Z_8\\ X_n^8 &= l\cdot X_{n-1}^8+s\cdot X_{n+1}^8+t\cdot Y_n^8+Z_8\\ Y_n^8 &= l\cdot Y_{n-1}^8+s\cdot Y_{n+1}^8+c_2(\tilde{d}_{k+1})\cdot X_{n+1}^8+Z_8\\ Z_8 &= \prod_{i=1}^{p}s_5(d_{k+i})\;\big|\;n\geq 1\,\}.\end{aligned}$$

We see at once from equations (117)–(120) and equation (121) that if we put for all $n \geq 0$

$$X_n^8 = \pi_J^p \circ \beta_1\big(d_k'(\tilde{d}_{k+1}^n)\big)$$
$$Y_n^8 = \pi_J^p \circ \beta_1\big(e_k'(\tilde{d}_{k+1}^n)\big),$$

that this will solve the system E_8. Let $C_8 = \{X_n^8, Y_n^8 : n \geq 0\}$. The set of exits is $U(C_8) = \{Z_8\}$. We may apply $CFAR^\infty$ and we find for all $n \geq 0$

$$\tau_{I'}(X_n^8) = \tau_{I'}(Y_n^8) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+i}).$$

A simple commutativity/absorption argument will yield that equation (130) is valid.

Now we will deduce the following result on a single *BC*-trace. For all $n \geq 0$

$$\left.\begin{array}{l}\pi_J^p \circ \tau_{I'}\big(b_k'(\tilde{d}_{k+1}^n)\big)\\ \pi_J^p \circ \tau_{I'}\big(c_k'(\tilde{d}_{k+1}^n)\big)\end{array}\right\} = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+i}). \tag{131}$$

To achieve this, we will need one more guarded recursive specification E_9.

$$\begin{aligned} E_9 = \big\{\, X_0^9 &= c_2(\tilde{d}_{k+1}) \cdot Y_1^9 \\ Y_0^9 &= t \cdot X_0^9 \\ X_n^9 &= l \cdot X_{n-1}^9 + s \cdot X_{n+1}^9 + c_2(\tilde{d}_{k+1}) \cdot Y_{n+1}^9 + Z_9 \\ Y_n^9 &= l \cdot Y_{n-1}^9 + s \cdot Y_{n+1}^9 + t \cdot X_n^9 + Z_9 \\ Z_9 &= c_3(\tilde{d}_{k+1}) \cdot \prod_{i=1}^{p} s_5(d_{k+i}) \;\big|\; n \geq 1 \,\big\}. \end{aligned}$$

Let $C_9 = \{X_n^9, Y_n^9 : n \geq 0\}$. It will be clear that this is a conservative cluster for E_9 in I'. The set of exits is $U(C_9) = \{Z_9\}$. So if we apply $CFAR^\infty$ we find for all $n \geq 0$

$$\tau_{I'}(X_n^9) = \tau_{I'}(Y_n^9) = \tau \cdot \prod_{i=1}^{p} s_5(d_{k+i}).$$

Let $\beta_8 = \beta\big(I', \{c_3(\tilde{d}_{k+1})\}\big)$. With the aid of equations (81), (82), (87) and (88) together with the just derived result on the *DE*-traces (130) it is not hard to see that if we put for all $n \geq 0$

$$X_n^9 = \pi_J^p \circ \beta_8\big(b_k'(\tilde{d}_{k+1}^n)\big)$$
$$Y_n^9 = \pi_J^p \circ \beta_8\big(c_k'(\tilde{d}_{k+1}^n)\big),$$

this is a solution for the guarded recursive specification E_9. So a straightforward calculation gives us that equation (131) is correct.

At this point we will make the final calculations to prove the theorem. Suppose that $k = 0$. We will show that equation (116) is correct. Let $p \geq 1$ be fixed and consider the deduction below.

$$\begin{aligned} \pi_J^p \circ \tau_{I'}\big(a_0'(\lambda)\big) &= \tau \cdot \pi_J^p \circ \tau_{I'}\big(b_0'(\lambda)\big) && \text{use (79)} \\ &= \tau \cdot \tau \cdot \prod_{i=1}^{p} s_5(d_i) && \text{because of (131)} \\ &= \tau \cdot \pi_J^p\Big(\prod_{i=1}^{\infty} s_5(d_i)\Big). \end{aligned}$$

In accord with *GIP* we may conclude that

$$\tau_{I'}\big(a_0'(\lambda)\big) = \tau \cdot \prod_{i=1}^{\infty} s_5(d_i),$$

which is exactly equation (116). We have also seen the relation between a_0' and A_0; see equation (115). And with the aid of this equation we conclude that

$$\tau_I \circ \partial_H\big(A_0(\lambda)\big) = \tau \cdot \prod_{i=1}^{\infty} s_5(d_i).$$

This ends the proof of (4.16).

5. Conclusions and further research

We have presented a theory in which it is possible to introduce linear unary operators and in which we can reason on these operators in a comfortable way. The theory that we developed is a generalization of the theory that is the subject of research in [**18**]. In there it was not yet possible to specify the projection operators. This is now incorporated in the theory. Moreover, the generalized induction principle has been added to the theory that we studied in [**18**], in which we already studied the two other principles *ODP* and *OSP*. The results of [**18**] can also be verified in this more general framework. Therefore, we emphasized in this paper on theorems concerning operators that could not be specified with the more restrictive form of the theory: the projection operators.

In section 4 we found some interesting applications on the subject of protocol verification. We showed in that section that *GIP* can be very useful in the area of protocol verification, especially for algebraic verification techniques, whereas *AIP* and its set of full projections is not very useful due to the fact that they do, in general, not commute with the abstraction operator.

We can still not specify the generalized state operator [**2**] with a linear functional specification so further research on this subject is recommended. A way how to achieve this is mentioned briefly in [**18**]. From section 4 it becomes clear that in this area also more work can be done: we verified three protocols that were just hypothetical ones. But we can see that it might be the case that we will not need the priority operator in the specification and verification of the *PAR*-protocol [**17**] anymore. Since this non-linear unary operator is used in there to calm down the time out. In our examples this time out can be executed at any moment and does not have to be tempered with the priority operator. We can make the three example protocols more symmetric by giving the two channel specifications the same behaviour; it will be not that hard to verify these protocols in the same way as we did for the asymmetric ones. Although it can be quite a lenghty job. It will also not be a difficult task to verify the examples with a stronger correctness criterion: namely that these protocols are, in fact, one bit buffers.

When we are able to specify the generalized state operator and more protocol verifications of this type have been investigated we can try to verify some sliding window protocols that have similar correctness criteria as the criterion that we used for our verifications.

Another interesting matter that can be worked out is the verification of theorem (4.12) and its more restrictive versions. In [**18**] we can already see such a theorem.

In this paper we have not mentioned anything on the model theoretic side of $ACP_{\tau,u}$. In [**18**] a model for the theory without abstraction has been given and this can be generalized effortlessly to the more general situation that we studied in this paper. However, on this subject still a lot of work has to be done.

If we look at the proofs that have been given in section 4 we can see that they consist of many steps that are almost the same. It might be the case that parts of these proofs can be done automagically and therefore it is also interesting to give formal specifications of these protocols with the use of an algebraic specification language. Examples of such languagese are *PSF* see [**15**] and μCRL see [**13**].

6. References

In this section we will catalogue all the references we made throughout this paper. We will refer to the first possible source in which an item can be found. This is not always the best source. General references to the subject of process algebra and its applications are [**3**] for the process algebra and [**1**] for several applications. In there the reader may find sometimes more convenient references.

[**1**] J. C. M. Baeten (Editor), *Applications of process algebra*, Cambridge tracts in theoretical computer science **17**, Cambridge university press, 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, W. P. Weijland, *Process algebra*, Cambridge tracts in theoretical computer science **18**, Cambridge university press, 1990.

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

[**5**] J. A. Bergstra, J. W. Klop, *Fair FIFO queues satisfy an algebraic criterion for protocol correctness*, CWI Report CS-R8405, Centre for mathematics and computer science, Amsterdam, 1984.

[**6**] 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.

[**7**] 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.

[**8**] 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.

[**9**] J. J. Brunekreef, *A formal specification of three sliding window protocols*, Report P9102, Programming Research Group, University of Amsterdam, 1991.

[**10**] R. J. van Glabbeek, *Bounded nondeterminism and the approximation induction principle in process algebra*, In: Proceedings STACS **87** (F.J. Brandenburg, G. Vidal-Naquet, M. Wirsing, eds.), LNCS **247**, Springer Verlag, pp. 336–347, 1987.

[**11**] R. J. van Glabbeek, F. W. Vaandrager, *Modular specifications in process algebra—with curious queues*, in: Algebraic Methods: Theory, Tools, and Applications (M. Wirsing, J. A. Bergstra, eds.), LNCS **394**, Springer Verlag, pp. 465–506, 1989.

[**12**] R. J. van Glabbeek, W. P. Weijland, *Branching time and abstraction in bisimulating semantics (extended abstract)*, Information Processing **89** 613–618, (G. X. Ritter, ed.), North Holland, Amsterdam 1989.

[**13**] J. F. Groote, A. Ponse, *The syntax and semantics of μCRL*, report CS-R9076, Centre for Mathematics and Computer Science, Amsterdam, 1990.

[**14**] G. Karjoth, *Stepwise specification of a sliding-window protocol by means of process algebra*, Proc. 1988 Intl. Zürich Seminar on Digital Communications (IEEE Catalog Nr. 88TH0202-2), 1988.

[**15**] S. Mauw, G. J. Veltink, *A process specification formalism*, Fundamenta Informaticae XIII, pp. 85–139, 1990.

[**16**] R. Milner, *A calculus of communicating systems*, LNCS **92**, Springer Verlag.

[**17**] F. W. Vaandrager, *Verification of two communication protocols by means of process algebra*, In [**1**].

[**18**] C. Verhoef, *An operator definition principle (for process algebras)*, Report P9105, Programming Research Group, University of Amsterdam, 1991.