A Complete Equational Axiomatization for BPA$\delta\epsilon$ with Prefix Iteration

Wan Fokkink & Hans Zantema
Utrecht University
Departments of Philosophy and of Computer Science
Utrecht, The Netherlands
fokkink@phil.ruu.nl hansz@cs.ruu.nl

Abstract

Prefix iteration $\mu^* x$ is added to Basic Process Algebra with deadlock and empty process. We present a finite equational axiomatization for this process algebra, and we prove that this axiomatization is complete with respect to strong bisimulation equivalence. This result is a mild generalization of a similar result in the setting of basic CCS in Fokkink (1994b).

To obtain this completeness result, we set up a rewrite system, based on the axioms. In order to prove that this rewrite system is terminating modulo AC of the $+$, we generalize a termination theorem from Zantema and Geser (1994) to the setting of rewriting modulo equations. Finally, we show that bisimilar normal forms are syntactically equal modulo AC of the $+$.

1 Introduction

Kleene (1956) defined a binary operator $\_^*\_$ in the context of finite automata, called *Kleene star* or *iteration*. Intuitively, the expression p^*q yields a solution for the recursive equation $X = p \cdot X + q$. In other words, p^*q can choose to execute either p, after which it evolves into p^*q again, or q, after which it terminates.

In this paper, we study the prefix variant of iteration in bisimulation equivalence, in the setting of Basic Process Algebra (BPA) together with the deadlock δ from Bergstra and Klop (1984) and the empty process ϵ from Koymans and Vrancken (1985).

Milner (1984) proposed an axiomatization for the Kleene star in a process algebra equivalent to BPA$_{\delta\epsilon}$, including a conditional axiom for iteration from Salomaa (1966), and he raised the question whether his axiomatization is complete with respect to bisimulation equivalence. This question is, to our knowledge, still open.

Bergstra, Bethke and Ponse (1994) considered the Kleene star in BPA, and they suggested a finite equational axiomatization for this algebra. Fokkink and Zantema (1994) proved that this axiomatization is complete with respect to strong bisimulation equivalence.

Sewell (1994) proved that there does not exist a complete finite equational axiomatization for the Kleene star in BPA$_\delta$ modulo bisimulation, due to equivalences

such as $(x^n)^*\delta \leftrightarrow x^*\delta$ for $n \geq 1$.[1] In order to obtain a complete finite equational axiomatization nevertheless, we replace the binary iteration operator x^*y by its unary prefix version μ^*x, where the argument at the left is restricted to atomic actions and deadlock and empty process. The resulting algebra is denoted by $\mathrm{BPA}^{p*}_{\delta\epsilon}$.

We propose five equational axioms for prefix iteration. Three of these axioms are actually instantiations of the three axioms for the Kleene star. We prove that these five axioms, together with the nine standard axioms of $\mathrm{BPA}_{\delta\epsilon}$, are a complete axiomatization for $\mathrm{BPA}^{p*}_{\delta\epsilon}$ with respect to bisimulation. This result has been inspired by the need for a complete axiomatization for prefix iteration in an extension of BPA_δ with discrete time, in a revision of Baeten and Bergstra (1992).

Our result is a mild generalization of a similar completeness result for basic CCS extended with prefix iteration in Fokkink (1994b), where multiplication is restricted to its prefix counterpart $a \cdot x$, and the empty process and single atomic actions in A are not used in the construction of the syntax. Besides the five extra axioms for $\mathrm{BPA}_{\delta\epsilon}$, compared with basic CCS, and two extra axioms to deal with terms of the form δ^*p and ϵ^*p, we only need one extra axiom for prefix iteration, namely

$$(a^*x)y \;=\; a^*(xy).$$

The strategy of the completeness proof that is presented here is fully different from the one in Fokkink (1994b). Process terms are modified by means of a rewrite system, and we show that bisimilar normal forms of this rewrite system are provably equal.

Proving termination of the rewrite system, which implies that each process term has at least one normal form, is an involved matter. We will apply an abstract commutation technique from Zantema and Geser (1994). This technique is closely related to the earlier technique of Bellegarde and Lescanne (1990). In order to apply this technique, first we have to generalize it to the setting of rewriting modulo equations. This generalization is presented in the Sections 3.2 and 3.3. It can be considered as a general applicable technique for proving termination of rewriting modulo equations. This is of interest itself, independent from the field of process algebra. Basically, termination of a rewrite system R is proved by means of termination of a simplified rewrite system S and an auxiliary rewrite system A connecting R and S. Surprisingly for extending this framework to the setting of rewriting modulo a set of equations E, no cooperation between R and E is required, only between A and E.

Aceto and Ingólfsdóttir (1995) study basic CCS with prefix iteration together with the silent step, modulo observation congruence. They extend the axiomatization from Fokkink (1994b) with two well-known equational axioms for abstraction, and with three new equational axioms which describe the interplay between abstraction and prefix iteration. They prove that their axiomatization for basic CCS with abstraction and prefix iteration is complete.

Fokkink (1995) studies $\mathrm{BPA}^{p*}_{\delta\epsilon}$ together with the silent step, modulo branching bisimulation. It turns out that two equational axioms suffice to describe the relation between the silent step and prefix iteration in that setting.

[1] Conjecture: neither does there exist a complete finite equational axiomatization for the Kleene star in BPA_ϵ modulo bisimulation, due to equivalences such as $((x+\epsilon)^n)^*y \leftrightarrow x^*y$ for $n \geq 1$.

Acknowledgements. Jos Baeten initiated this research, and Luca Aceto and Alfons Geser provided helpful comments.

2 BPA$\delta\epsilon$ with Prefix Iteration

We assume an alphabet A of atomic actions, and two special constants δ and ϵ, which represents deadlock and empty process respectively. We use a to range over A and μ to range over $A \cup \{\delta, \epsilon\}$. The signature of the process algebra $\mathrm{BPA}^{p*}_{\delta\epsilon}$ is built from the constants in $A \cup \{\delta, \epsilon\}$, alternative composition $x + y$, sequential composition $x \cdot y$, and prefix iteration $\mu^* x$.

Table 1 presents an operational semantics for $\mathrm{BPA}^{p*}_{\delta\epsilon}$ in the style of Plotkin (1981). Prefix iteration $a^* x$ can choose to execute either a, after which it evolves into $a^* x$ again, or x. The expression $x\downarrow$ denotes successful termination of x.

$$\epsilon\downarrow \qquad a \xrightarrow{a} \epsilon$$

$$\frac{x\downarrow}{(x+y)\downarrow \quad (y+x)\downarrow} \qquad \frac{x \xrightarrow{a} x'}{x+y \xrightarrow{a} x' \quad y+x \xrightarrow{a} x'}$$

$$\frac{x\downarrow \quad y\downarrow}{(x\cdot y)\downarrow} \qquad \frac{x\downarrow \quad y \xrightarrow{a} y'}{x\cdot y \xrightarrow{a} y'} \qquad \frac{x \xrightarrow{a} x'}{x\cdot y \xrightarrow{a} x'\cdot y}$$

$$\frac{x\downarrow}{(\mu^* x)\downarrow} \qquad \frac{x \xrightarrow{a} x'}{\mu^* x \xrightarrow{a} x'} \qquad a^* x \xrightarrow{a} a^* x$$

Table 1: Action rules for $\mathrm{BPA}^{p*}_{\delta\epsilon}$

Our model for $\mathrm{BPA}^{p*}_{\delta\epsilon}$ consists of all the closed terms that can be constructed from the constants in $A \cup \{\delta, \epsilon\}$ together with the three operators. That is, the BNF grammar for the collection of process terms is as follows:

$$p \quad ::= \quad \mu \mid p + p \mid p \cdot p \mid \mu^* p.$$

As binding convention, * binds stronger than $\cdot$, which binds stronger than $+$. Often, $p \cdot q$ will be abbreviated to pq.

Process terms are considered modulo *(strong) bisimulation equivalence* from Park (1981). Intuitively, two process terms are bisimilar if they have the same branching structure.

Definition 2.1 *Two processes p and q are* bisimilar, *denoted by $p \leftrightarrows q$, if there exists a symmetric binary relation $\mathcal{B}$ on processes, which relates p and q, such that*

1. *if $r \xrightarrow{a} r'$ and $r\mathcal{B}s$, then there is a transition $s \xrightarrow{a} s'$ with $r'\mathcal{B}s'$,*

2. if $r\downarrow$ and $r\mathcal{B}s$, then $s\downarrow$.

The action rules in Table 1 are in the 'path' format of Baeten and Verhoef (1993). Hence, bisimulation equivalence is a congruence with respect to all the operators, i.e. if $p \underline{\leftrightarrow} p'$ and $q \underline{\leftrightarrow} q'$, then $p+q \underline{\leftrightarrow} p'+q'$ and $p \cdot q \underline{\leftrightarrow} p' \cdot q'$ and $\mu^* p \underline{\leftrightarrow} \mu^* p'$. See Baeten and Verhoef (1993) for the definition of the path format, and for a proof of this congruence result. (This proof uses the extra assumption that the action rules are 'well-founded'. In Fokkink (1994a) it has been shown that this requirement can be dropped.)

Furthermore, the action rules for $\mathrm{BPA}_{\delta\epsilon}$ are 'pure', which is a syntactic criterion from Groote and Vaandrager (1992), and the three action rules for prefix iteration incorporate the Kleene star in the left-hand side of their conclusions. Hence, $\mathrm{BPA}^{p*}_{\delta\epsilon}$ is an operationally conservative extension of $\mathrm{BPA}_{\delta\epsilon}$, i.e. the action rules for prefix iteration do not influence the transition systems of $\mathrm{BPA}_{\delta\epsilon}$ terms. See Verhoef (1994) for a proof of this conservativity result.

Table 2 contains an axiom system for $\mathrm{BPA}^{p*}_{\delta\epsilon}$, which consists of the nine axioms from $\mathrm{BPA}_{\delta\epsilon}$ together with five axioms for prefix iteration. The axioms MI1,3 already appeared in Hennessy (1981), as axioms (in CCS) for the delay operator, which is an instance of prefix iteration.

A1	$x+y$	$=$	$y+x$
A2	$(x+y)+z$	$=$	$x+(y+z)$
A3	$x+x$	$=$	x
A4	$(x+y)z$	$=$	$xz+yz$
A5	$(xy)z$	$=$	$x(yz)$
A6	$x+\delta$	$=$	x
A7	δx	$=$	δ
A8	$x\epsilon$	$=$	x
A9	ϵx	$=$	x
MI1	$a \cdot a^* x + x$	$=$	$a^* x$
MI2	$(a^* x)y$	$=$	$a^*(xy)$
MI3	$a^*(a^* x)$	$=$	$a^* x$
MI4	$\delta^* x$	$=$	x
MI5	$\epsilon^* x$	$=$	x

Table 2: Axioms for $\mathrm{BPA}^{p*}_{\delta\epsilon}$

In the sequel, $p=q$ will mean that this equality can be derived from the axioms. The axiomatization A1-9+MI1-5 is sound with respect to bisimulation equivalence, i.e. if $p=q$ then $p \underline{\leftrightarrow} q$. Since bisimulation is a congruence, this can be verified by checking soundness for each axiom separately, which is left to the reader. In this

paper it is proved that the axiomatization is complete with respect to bisimulation, i.e. if $p \leftrightarrows q$ then $p = q$.

3 Construction of Normal Forms

3.1 A rewrite system

From now on, process terms are considered modulo AC of the $+$, that is, modulo associativity and commutativity of the $+$. In the sequel, $p =_{\mathrm{AC}} q$ denotes that p and q are equal modulo AC of the $+$, and we say that p and q are of the same form.

1.	$(x+y)z$	$\longrightarrow$	$xz+yz$
2.	$(xy)z$	$\longrightarrow$	$x(yz)$
3.	δx	$\longrightarrow$	δ
4.	ϵx	$\longrightarrow$	x
5.	$(a^*x)y$	$\longrightarrow$	$a^*(xy)$
6.	δ^*x	$\longrightarrow$	x
7.	ϵ^*x	$\longrightarrow$	x
8.	a^*x+y	$\longrightarrow$	$a\cdot a^*x+x+y$
9.	$a^*(b^*x)$	$\longrightarrow$	$a^*(b\cdot b^*x+x)$

Table 3: The rewrite system R_0

Table 3 contains a rewrite system R_0, which reduces sequential composition to its prefix counterpart (rules 1-5), and which eliminates expressions of the form δ^*x and ϵ^*x (rules 6 and 7), and which reduces occurrences of prefix iteration in the context of alternative composition and of prefix iteration (rules 8 and 9). The rewrite rules are to be interpreted modulo AC of the $+$.

In the completeness proof it will be shown that bisimilar normal forms of R_0 are provably equal. Hence, in order to obtain completeness for the full class of process terms, we desire to know that R_0 reduces each process term to a normal form, which does not reduce any further. We shall prove a stronger fact, namely that R_0 is terminating modulo AC of the $+$, which means that R_0 does not allow any infinite reductions.

The rewrite rules 8 and 9 are self-embedding: their left-hand sides can be embedded in the corresponding right-hand sides. Hence, it is not possible to prove termination of R_0 by means of a weight function in the natural numbers, see e.g. Zantema (1994). Termination of rules 8 and 9 alone can be proved elegantly by means of the technique of dummy elimination from Ferreira and Zantema (1994), because sequential composition occurs only at the right-hand side of these two rewrite rules. However, this technique cannot be applied to the full rewrite system R_0.

In order to prove termination of R_0 modulo AC of the $+$, we apply a technique from Zantema and Geser (1994), based on an abstract commutation criterion. Since

we will need that result in a more general setting, namely rewriting modulo equations, in Section 3.2 we generalize the abstract commutation criterion accordingly. Next, in Section 3.3 we describe how this applies in rewriting. Finally, in Section 3.4 we use this technique to prove termination of R_0 modulo AC of the $+$.

3.2 Abstract termination

The proof that the rewrite system R_0 is terminating modulo AC of the $+$, which is presented in Section 3.4, is based on a theorem that can be given in a very general abstract setting. This theorem, which is a generalization of a result from Zantema and Geser (1994) to the setting of rewriting modulo equations, will be presented and proved in this section.

Let R, S, T, E denote binary relations on a fixed set $\mathcal{V}$. We write a dot symbol for relational composition, i.e. one has $t(R.S)t'$ if and only if there exists t'' such that tRt'' and $t''St'$. We write R^+ for the transitive closure of R and R^* for the reflexive transitive closure of R. Further we write $R \subseteq S$ if tRt' implies tSt'. Clearly, if $R \subseteq S$ then $R.T \subseteq S.T$ and $T.R \subseteq T.S$.

We write $\infty(t, R)$ if there exists an infinite sequence $t_1, t_2, t_3, \ldots$ such that $t = t_1$ and $t_i R t_{i+1}$ for all $i = 1, 2, 3, \ldots$. A relation R is called *terminating* if there does not exist any term t satisfying $\infty(t, R)$.

In the following lemma we collect some standard properties for relations, which are easy to check.

Lemma 3.1

1. *If $R.S \subseteq S^*.R$, then $R.S^* \subseteq S^*.R$.*
2. *If $R.S \subseteq S.R^*$, then $R^*.S \subseteq S.R^*$.*
3. *If $R.S \subseteq T^+.R$ and $t'Rt$ and $\infty(t, S)$, then $\infty(t', T)$.*

For relations R, E we write R/E for $E^*.R.E^*$. The intuition here is that the reduction relation R is taken modulo equations E. However, we do not need that E is symmetric, hence our theorem is even on relative termination which is more general than modulo an equivalence. Now we state and prove our abstract termination theorem.

Theorem 3.2 *Let R, S, T, E be binary relations satisfying*

1. *S/E is terminating,*
2. *$R \subseteq S^+.T^*$,*
3. *$T.R \subseteq R^+.T^*$,*
4. *$T.E \subseteq E^*.T$.*

Then R/E is terminating.

Proof. From condition 4 and the first item of Lemma 3.1 we conclude $T.E^* \subseteq E^*.T$. From this and condition 3 we conclude:

$$\begin{array}{rcl}(T/E).(R/E) & = & E^*.T.E^*.R.E^* \\ & \subseteq & E^*.E^*.T.R.E^* \\ & = & E^*.T.R.E^* \\ & \subseteq & E^*.R^+.T^*.E^* \\ & \subseteq & E^*.R^+.(R \cup T)^*.E^* \\ & = & E^*.R.(R \cup T)^*.E^* \\ & \subseteq & (R/E).((R \cup T)/E)^*.\end{array}$$

Since also $(R/E).(R/E) \subseteq (R/E).((R \cup T)/E)^*$, we obtain

$$((R \cup T)/E).(R/E) \;=\; (R/E).(R/E) \cup (T/E).(R/E) \;\subseteq\; (R/E).((R \cup T)/E)^*.$$

From the second item of Lemma 3.1 and condition 2 we conclude

$$\begin{array}{rcl}((R \cup T)/E)^*.(R/E) & \subseteq & (R/E).((R \cup T)/E)^* \\ & = & E^*.R.E^*.((R \cup T)/E)^* \\ & \subseteq & E^*.S^+.T^*.E^*.((R \cup T)/E)^* \\ & = & E^*.S^+.((R \cup T)/E)^* \\ & \subseteq & (S/E)^+.((R \cup T)/E)^*\end{array}$$

Assume that R/E does not terminate. Then there exists an element t with $\infty(t, R/E)$. Clearly $t((R \cup T)/E)^*t$, hence the third item of Lemma 3.1 yields $\infty(t, S/E)$. This contradicts condition 1. □

To stress the subtlety of this theorem, we show that condition 4 may not be weakened to $T.E \subseteq E^*.T^+$.

Example 3.3 *Let* $\mathcal{V} = \{1, 2, 3, 4\}$ *and*

$$\begin{array}{llllll} 1R3, & & & & & \\ 1S4, & & & & & \\ 4T3 & 3T2 & 1T1 & 2T2 & 3T3, & \\ 1E2 & 2E1 & 2E3 & 3E2. & & \end{array}$$

S/E is terminating, because S consists only of $1S4$, and 4 cannot be reduced by S nor by E. The relation inclusions in conditions 2 and 3 in Theorem 3.2 and $T.E \subseteq E^*.T^+$ are easily checked.

$$\begin{array}{rll} R \subseteq S^+.T^*: & 1R3 & 1S4T3. \\ & & \\ T.R \subseteq R^+.T^*: & 1T1R3 & 1R3. \\ & & \\ T.E \subseteq E^*.T^+: & 4T3E2 & 4T3T2 \\ & 3T2E1 & 3E2E1T1 \\ & 3T2E3 & 3T3 \\ & 1T1E2 & 1E2T2 \\ & 2T2E1 & 2E1T1 \\ & 2T2E3 & 2E3T3 \\ & 3T3E2 & 3T2. \end{array}$$

However, R/E is not terminating: $1R3E2E1R3\cdots$.

3.3 Application to rewrite systems

Before applying Theorem 3.2 to rewrite systems, first we recall some standard terminology from term rewriting. See e.g. Klop (1992) for an overview of the field of term rewriting.

Definition 3.4

- *A rewrite rule $l \longrightarrow r$ is called* left-linear *if each variable occurs at most once in l.*
- *A rewrite rule $l \longrightarrow r$ is called* non-erasing *if each variable in l also occurs in r.*

A rewrite system is called left-linear *or* non-erasing*, respectively, if all its rules are so.*

An equation $l = r$ is called linear *if both $l \longrightarrow r$ and $r \longrightarrow l$ are left-linear. An equation $l = r$ is called* non-erasing *if both $l \longrightarrow r$ and $r \longrightarrow l$ are so. A system of equations is called* linear *or* non-erasing*, respectively, if all its equations are so.*

In particular, both commutativity and associativity are linear and non-erasing.

Theorem 3.2 can be applied to prove termination of rewrite systems modulo equations. Then for R one chooses the rewrite relation of a rewrite system (also called R) for which termination has to be proved modulo some equations. For E one chooses the 'one-step equalities' corresponding to these equations. For S one chooses an adaptation of R for which termination modulo the equations can be proved. Then condition 1 of Theorem 3.2 is fulfilled. For each rule $l \to r$ of R there is to be a rule $l \to r'$ in S such that $r \to_A^* r'$ for some auxiliary rewrite system A. For the relation T one chooses the inverse of the rewrite relation of A, so that condition 2 of Theorem 3.2 is also fulfilled. Now condition 3 reads:

> if $t \to_A t'$ and $t \to_R t''$, then there exists a u for which $t' \to_R^+ u$ and $t'' \to_A^* u$.

If A is left-linear and non-erasing, and if R is left-linear, then this requirement is always fulfilled for non-overlapping redexes. Hence, this condition can be verified by a finite analysis of overlapping redexes. In the typical case, in the first attempt for A the condition does not hold, and A has to be extended a number of times to obtain condition 3. This is a kind of *completion*, similar to what is done in Bellegarde and Lescanne (1990). In the application of Theorem 3.2 in this paper, the rewrite systems R and S are also extended during this completion, and the final auxiliary rewrite system A has infinitely many rules.

Finally, for the system E of equations, condition 4 of Theorem 3.2 reads:

> if $t \to_A t'$ and $t \leftrightarrow_E t''$, then there exists a u for which $t' =_E u$ and $t'' \to_A u$.

Here $\leftrightarrow_E$ denotes 'one-step equalities' corresponding to E, while $=_E$ denotes $\leftrightarrow_E^*$, being the generated congruence. If E is linear and non-erasing, and if A is left-linear, then this condition can be verified by a finite analysis of overlapping redexes, similar as for condition 3.

3.4 Termination of the rewrite system R_0

The intuition behind the termination proof of R_0 is that the expansion from a pattern a^*p to $a \cdot a^*p + p$, as is done by rules 8 and 9, can occur at most only once for every occurrence of a prefix iteration symbol. We formalize this as follows. Extend the signature with unary function symbols $a^\#\_$ for $a \in A$. Intuitively, these new function symbols will be used to register that the expansion from a^*p to $a \cdot a^*p + p$ has been done. In a first attempt to apply Theorem 3.2, we choose R to be R_0, and S to be the simplified variant of R in which the patterns $a \cdot a^*p + p$ in the right-hand sides of rules 8 and 9 have been replaced by $a \cdot a^\#p + p$. Furthermore, we choose E to be the AC rules for $+$. It is not difficult to see that S/E is terminating. As a first obvious try, we choose A to consist of the rule

$$a(a^*x) \quad \longrightarrow \quad a(a^\#x).$$

Now condition 2 is easily checked, but condition 3 does not yet hold. Therefore we extend the systems A, R and S with some new rules, triggered by the desired validity of condition 3. This process of completion ends in the following choices for the systems A, R and S.

For the rewrite system R we choose the original system R_0 extended by the following two new rules:

$$\begin{array}{rrcl} 10. & (a^\#x)y & \longrightarrow & a^\#(xy) \\ 11. & a^\#(b^*x) & \longrightarrow & a^\#(b \cdot b^*x + x). \end{array}$$

Since rewrite rules are applied modulo AC of the $+$, we take $E = \{x + y = y + x,\ (x + y) + z = x + (y + z)\}$. We shall apply Theorem 3.2 yielding termination of R/E, which immediately implies termination of R_0 modulo AC of the $+$.

The rewrite system S is obtained by a slight modification of R in which the patterns $a \cdot a^*p + p$ as they appear in the right-hand sides of rules 8, 9 and 11 have been replaced by $a \cdot a^\#p + p$. That is, S consists of the rules 1-7, 10 and

$$\begin{array}{rrcl} 8'. & a^*x + y & \longrightarrow & a \cdot a^\#x + x + y \\ 9'. & a^*(b^*x) & \longrightarrow & a^*(b \cdot b^\#x + x) \\ 11'. & a^\#(b^*x) & \longrightarrow & a^\#(b \cdot b^\#x + x). \end{array}$$

Finally, we define the rewrite system A to be the following infinite collection of rewrite rules:

$$\begin{array}{rrcl} r_0 & a(a^*x) & \longrightarrow & a(a^\#x) \\ r_1 & a((a^*x)y_0) & \longrightarrow & a((a^\#x)y_0) \\ r_2 & a(((a^*x)y_0)y_1) & \longrightarrow & a(((a^\#x)y_0)y_1) \\ & \vdots & & \end{array}$$

More precisely, A consists of rewrite rules r_i of the form $a \cdot C_i[a^*x] \longrightarrow a \cdot C_i[a^\#x]$ for $i \geq 0$, where the contexts $C_i[]$ are defined inductively by

$$C_0[] = [], \qquad\qquad C_{i+1}[] = C_i[] \cdot y_i,$$

with y_i a fresh variable. Equivalently, one can say that r_i is of the form $a \cdot D_i[a^*x] \longrightarrow a \cdot D_i[a^\#x]$, where the contexts $D_i[]$ are defined inductively by

$$D_0[] = [], \qquad D_{i+1}[] = D_i[[] \cdot z_i],$$

with z_i a fresh variable. We will need both representations of r_i later on.

Now we verify the four conditions of Theorem 3.2.

1. S/E is terminating.

   Define the following weight function on terms.

$$\begin{array}{rcl} w(\mu) & = & 2 \\ w(p+q) & = & w(p) + w(q) \\ w(pq) & = & w(p)^2 w(q) \\ w(a^*p) & = & 5w(p) + 5 \\ w(a^\#p) & = & w(p) + 1 \end{array}$$

   Note that terms which are equal modulo AC of the $+$ have the same weight. It is easy to see that the weight of terms strictly decreases under application of rules in S. Hence, S/E is terminating.

2. For each rule $l \to r$ of R there is a rule $l \to r'$ in S such that $r \to_A^* r'$.

   Only rules 8, 9 and 11 in R have been adapted in S. For these three rules in R, the rule r_0 in A, $a(a^*x) \longrightarrow a(a^\#x)$ can be applied to obtain the corresponding right-hand sides in S.

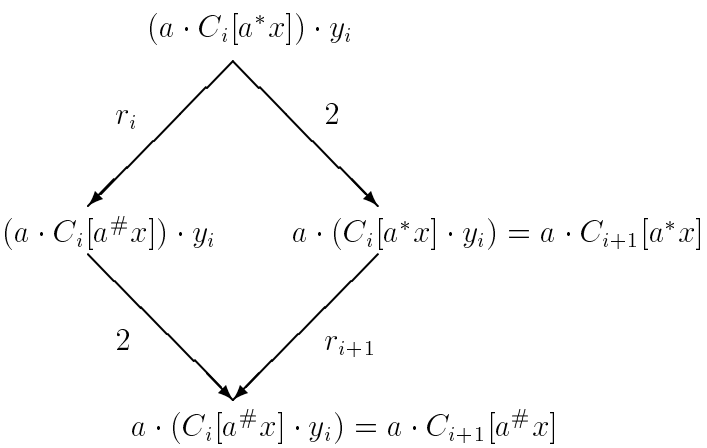
   The other rules in R and S coincide, so for those rules we can take r for r'.

3. If $t \to_A t'$ and $t \to_R t''$, then there exists a u for which $t' \to_R^+ u$ and $t'' \to_A^* u$.

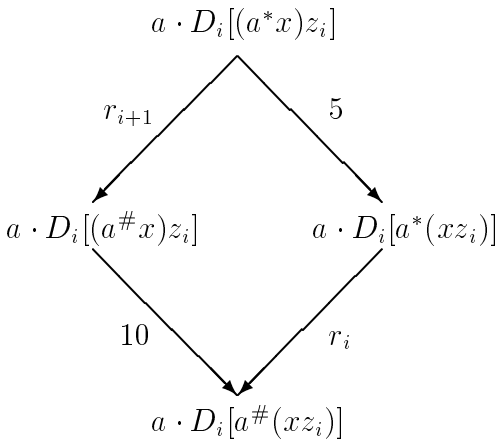
   Note that A is left-linear and non-erasing, and that R is left-linear.

   A straightforward analysis of overlapping redexes learns that there are three types of overlaps between a left-hand side of A and a left-hand side of R, which involve rules 2, 5 and 9 in R respectively. We treat these three cases separately.

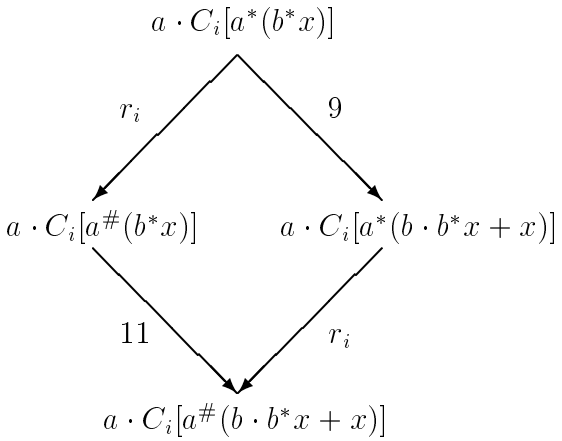
   (a) A term $(a \cdot C_i[a^*x]) \cdot y_i$ can be reduced by rule r_i in A and by rule 2 in R. This overlapping redex is convergent, owing to rule r_{i+1} in A.



(b) A term $a \cdot D_i[(a^*x) \cdot z_i]$ can be reduced by rule r_{i+1} in A and by rule 5 in R. This overlapping redex is convergent, owing to rule 10 in R.



(c) A term $a \cdot C_i[a^*(b^*x)]$ can be reduced by rule r_i in A and by rule 9 in R. This overlapping redex is convergent, owing to rule 11 in R.



4. If $t \to_A t'$ and $t \leftrightarrow_E t''$, then there exists a u for which $t' =_E u$ and $t'' \to_A u$.

   Since E is linear and non-erasing and A is left-linear, this holds for all non-overlapping redexes. Since all left-hand and right-hand sides of E contain no other symbols than $+$, and the left-hand sides of A contain no $+$ symbols, no overlapping redexes are possible.

So according to Theorem 3.2, we may conclude that R/E is terminating. Hence, the rewrite system R_0 in Table 3 is terminating modulo AC of the $+$.

4 Completeness of the Axioms

In this section we present the proof of the completeness theorem for the axioms of $\mathrm{BPA}^{p*}_{\delta\epsilon}$, with respect to bisimulation equivalence.

4.1 Basic terms

For convenience, we adapt normal forms a bit further to *basic terms*. In the completeness proof it will be shown that bisimilar basic terms are provably equal.

In the sequel, ξ ranges over the set $\{\delta, \epsilon\}$.

Definition 4.1 *A* basic term *is of the form either* $\sum_{i=1}^{n} a_i p_i + \xi$ *or* $a^*(\sum_{i=1}^{n} a_i p_i + \xi)$*, where the terms* p_i *are basic.*

Lemma 4.2 *Each process term is provably equal to a basic term.*

Proof. R_0 is terminating modulo AC of the $+$, so each process term has at least one normal form of R_0. Since each of the rewrite rules can be deduced from the axioms, each process term is provably equal to its normal forms.

Assume a normal form q. It follows from the rewrite rules in R_0 that q is a sum of terms of the form μ or aq' or a^*q', with q' a normal form, where a^*q' does not occur as an argument of alternative composition nor of prefix iteration. In other words, q is of the form either $\sum_{i=1}^{k} a_i q_i + \sum_{j=1}^{l} \mu_j$ or $a^*(\sum_{i=1}^{k} a_i q_i + \sum_{j=1}^{l} \mu_j)$, where the terms q_i are normal forms. By induction on size, i.e. on the number of function symbols in q, we may assume that the normal forms q_i are provably equal to basic terms.

The normal form q can be adapted to a basic term, by means of the axioms, as follows. Remove all occurrences of summands δ (axiom A6), remove double occurrences of summands ϵ (axiom A3), add one summand δ if there is no summand ϵ present (axiom A6), and replace summands a by $a\epsilon$ (axiom A8).

Hence, each normal form of R_0 is provably equal to a basic term. Since each process terms is provably equal to a normal form, it follows that each process term is provably equal to a basic term. □

4.2 The completeness theorem

The following lemma stems from Aceto and Ingólfsdóttir (in the setting of CCS).

Lemma 4.3 *If* $a^*p \leftrightarrows b^*q$*, then* $a = b$*.*

Proof sketch. If $a^*p \leftrightarrows b^*q$, then it follows that a^*p exhibits the infinite trace of actions $(ab)^\omega$. Thus, this lemma is an immediate consequence of the following fact.

- If $p_n \xrightarrow{a_n} p_{n+1}$ for $n = 0, 1, 2, ...$, then there is an N such that $a_n = a_N$ for $n > N$.

The proof of this fact is an easy exercise by structural induction on terms, which is left to the reader. □

Theorem 4.4 *The axiomatization* A1-9 + MI1-5 *for* $\mathrm{BPA}^{p*}_{\delta\epsilon}$ *is complete with respect to bisimulation equivalence.*

Proof. Since each process term is provably equal to a basic term, it is sufficient to show that bisimilar basic terms are provably equal. We deduce the following statement I_n by induction on n.

I_n If p and q are bisimilar basic terms with $size(p) + size(q) = n$, then $p = q$.

Suppose that we have already proved I_n for $n < N$, and let p and q are bisimilar basic terms with $size(p) + size(q) = N$. We prove that $p = q$, in three distinct cases, which distinguish the possible forms of p and q.

1. $p =_{\mathrm{AC}} a^*(\sum_i a_i p_i + \xi)$ and $q =_{\mathrm{AC}} \sum_j b_j q_j + \xi'$.

   $p \leftrightarrows q$, so $p \downarrow$ if and only if $q \downarrow$. Hence, ξ represents ϵ if and only if ξ' represents ϵ, so $\xi = \xi'$.

   Since $p \xrightarrow{a} p$, and since $p \leftrightarrows q$, there is a transition $q \xrightarrow{a} q'$ where $p \leftrightarrows q'$. Thus, $b_k = a$ and $p \leftrightarrows q_k$ for some k. Since $size(q_k) < size(q)$, the induction base I_{N-1} yields $p = q_k$. Hence, $ap = b_k q_k$.

   Since $p \xrightarrow{a_i} p_i$, and since $p \leftrightarrows q$, there is a transition $q \xrightarrow{a_i} q'$ where $p_i \leftrightarrows q'$. Hence, $b_k = a_i$ and $p_i \leftrightarrows q_k$ for some k. By induction $p_i = q_k$, so $a_i p_i = b_k q_k$.

   Thus, each summand of $ap + \sum_i a_i p_i + \xi$ is provably equal to a summand of q. By the symmetric argument, we find that each summand of q is provably equal to a summand of $ap + \sum_i a_i p_i + \xi$. Hence, $p \overset{\mathrm{MI1}}{=} ap + \sum_i a_i p_i + \xi = q$.

2. $p =_{\mathrm{AC}} \sum_i a_i p_i + \xi$ and $q =_{\mathrm{AC}} \sum_j b_j q_j + \xi'$.

   In this case, we can repeat the argument of the previous case to find that each summand of p is provably equal to a summand of q, and vice versa. Hence, $p = q$.

3. $p =_{\mathrm{AC}} a^*(\sum_{i \in I} a_i p_i + \xi)$ and $q =_{\mathrm{AC}} b^*(\sum_{j \in J} b_j q_j + \xi')$.

   By symmetry, we may assume that $size(p) \geq size(q)$. Since $p \leftrightarrows q$, Lemma 4.3 yields $a = b$.

   $p \leftrightarrows q$, so $p \downarrow$ if and only if $q \downarrow$. Hence, ξ represents ϵ if and only if ξ' represents ϵ, so $\xi = \xi'$.

   We distinguish three cases.

   - $q_j \leftrightarrows p$ for some $j \in J$.
     Induction yields $q_j = p$. Moreover, $q_j \leftrightarrows p \leftrightarrows q$ and $size(q) \leq size(p)$, so induction yields $q_j = q$. Hence, $p = q_j = q$.
   - For each $i \in I$, $a_i \neq a$ or $p_i \not\leftrightarrows q$, and for each $j \in J$, $q_j \not\leftrightarrows p$.
     In this case, each transition $p \xrightarrow{a_i} p_i$ of p, for $i \in I$, can only be mimicked by a transition $q \xrightarrow{b_j} q_j$ of q for some $j \in J$, and vice versa. So by induction each summand $a_i p_i$ for $i \in I$ is provably equal to a summand $b_j q_j$ for $j \in J$, and vice versa. Hence, $p =_{\mathrm{AC}} a^*(\sum_{i \in I} a_i p_i + \xi) = a^*(\sum_{j \in J} b_j q_j + \xi') =_{\mathrm{AC}} q$.
   - $p_i \leftrightarrows q$ and $a_i = a$ for some $i \in I$, and for each $j \in J$, $q_j \not\leftrightarrows p$.
     Abbreviate $\sum_{j \in J} b_j q_j + \xi'$ to q'.
     Let I_0 be the non-empty subset of elements $i \in I$ for which $p_i \leftrightarrows q$ and $a_i = a$. For $i \in I_0$, induction yields $p_i = q$, so $a_i p_i = aq =_{\mathrm{AC}} a \cdot a^* q'$. Hence, $\sum_{i \in I} a_i p_i = a \cdot a^* q' + \sum_{i \in I \setminus I_0} p_i$.

Each transition $p \xrightarrow{a_i} p_i$ of p for $i \in I \backslash I_0$ can only be mimicked by a transition $q \xrightarrow{b_j} q_j$ of q for some $j \in J$, and vice versa. So by induction each summand $a_i p_i$ for $i \in I \backslash I_0$ is provably equal to a summand $b_j q_j$ for $j \in J$, and vice versa. Hence, $p =_{\mathrm{AC}} a^*(\sum_{i \in I_0} a_i p_i + \sum_{i \in I \backslash I_0} a_i p_i + \xi) = a^*(a \cdot a^* q' + q') \stackrel{\mathrm{MI1}}{=} a^*(a^* q') \stackrel{\mathrm{MI4}}{=} a^* q' =_{\mathrm{AC}} q$. $\Box$

4.3 Complete axiomatizations for subalgebras

With a similar proof scheme as has been used for the completeness proof for $\mathrm{BPA}^{p*}_{\delta\epsilon}$, with the cases for δ and/or ϵ omitted, we obtained the following results.

Theorem 4.5 *The axiomatization* A1-5 + MI1-3 *for* BPA^{p*} *is complete with respect to bisimulation equivalence.*

Theorem 4.6 *The axiomatization* A1-7 + MI1-4 *for* $\mathrm{BPA}^{p*}_{\delta}$ *is complete with respect to bisimulation equivalence.*

Theorem 4.7 *The axiomatization* A1-5,8,9 + MI1-3,5 *for* $\mathrm{BPA}^{p*}_{\epsilon}$ *is complete with respect to bisimulation equivalence.*

References

[1] Aceto, L., and Ingólfsdóttir, A. (1995), A complete equational axiomatization for prefix iteration with silent steps, Report RS-95-5, University of Aarhus.

[2] Baeten, J. C. M., and Bergstra, J. A. (1992), Discrete time process algebra, Report P9208, University of Amsterdam. [To appear in *Formal Aspects of Computing*.]

[3] Baeten, J. C. M., and Verhoef, C. (1993), A congruence theorem for structured operational semantics with predicates, *in* "Proceedings, 4th Conference on Concurrency Theory (CONCUR'93), Hildesheim," (E. Best, ed.), pp. 477–492, Lecture Notes in Computer Science, Vol. 715, Springer-Verlag.

[4] Bellegarde, F., and Lescanne, P. (1990), Termination by completion, *Applicable Algebra in Engineering, Communication and Computation*, 1:79–96.

[5] Bergstra, J. A., Bethke, I., and Ponse, A. (1994), Process algebra with iteration and nesting, *The Computer Journal*, 37(4):243–258.

[6] Bergstra, J. A., and Klop, J. W. (1984), Process algebra for synchronous communication. *Information and Control*, 60(1/3):109–137.

[7] Ferreira, M. C. F., and Zantema, H. (1994), Dummy elimination: making termination easier, Report UU-CS-1994-47, Utrecht University.

[8] Fokkink, W. J. (1994a), The tyft/tyxt format reduces to tree rules, *in* "Proceedings, 2nd Symposium on Theoretical Aspects of Computer Software (TACS'94), Sendai, Japan," (M. Hagiya and J.C. Mitchell, eds.), pp. 440–453, Lecture Notes in Computer Science, Vol. 789, Springer-Verlag.

[9] Fokkink, W. J. (1994b), A complete equational axiomatization for prefix iteration, *Information Processing Letters*, 52(6):333–337.

[10] Fokkink, W. J. (1995), A complete axiomatization for prefix iteration in branching bisimulation, Logic Group Preprint Series 126, Utrecht University.

[11] Fokkink, W. J., and Zantema, H. (1994), Basic process algebra with iteration: completeness of its equational axioms, *The Computer Journal*, 37(4):259–267.

[12] Groote, J. F., and Vaandrager, F. W. (1992), Structured operational semantics and bisimulation as a congruence, *Information and Computation*, 100(2):202–260.

[13] Hennessy, M. (1981), A term model for synchronous processes, *Information and Control*, 51(1):58–75.

[14] Kleene, S. C. (1956), Representation of events in nerve nets and finite automata, *in* "Automata Studies," pp. 3–41, Princeton University Press.

[15] Klop, J. W. (1992), Term rewriting systems, *in* "Handbook of Logic in Computer Science, Volume I, Background: Computational Structures," (S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum, eds.), pp. 1–116, Oxford University Press.

[16] Koymans, C. J. P., and Vrancken, J. L. M. (1985), Extending process algebra with the empty process ϵ, Logic Group Preprint Series 1, University of Utrecht.

[17] Milner, R. (1984), A complete inference system for a class of regular behaviours, *Journal of Computer and System Sciences*, 28:439–466.

[18] Park, D. M. R. (1981), Concurrency and automata on infinite sequences, *in* "Proceedings, 5th GI Conference," (P. Deussen, ed.), pp. 167–183, Lecture Notes in Computer Science, Vol. 104, Springer-Verlag.

[19] Plotkin, G. D. (1981), A structural approach to operational semantics, Report DAIMI FN-19, Aarhus University.

[20] Salomaa, A. (1966), Two complete axiom systems for the algebra of regular events, *Journal of the ACM*, 13(1):158–169.

[21] Sewell, P. (1994), Bisimulation is not finitely (first order) equationally axiomatisable, *in* "Proceedings, 9th IEEE Symposium on Logic in Computer Science (LICS'94), Paris," pp. 62–70, IEEE Computer Society Press.

[22] Verhoef, C. (1994), A general conservative extension theorem in process algebra, *in* "Proceedings, IFIP Conference on Programming Concepts, Methods and Calculi (PROCOMET'94), San Miniato," (E.-R. Olderog, ed.), pp. 149–168, IFIP Transactions A-56, Elsevier.

[23] Zantema, H. (1994), Termination of term rewriting: interpretation and type elimination, *Journal of Symbolic Computation*, 17(1):23–50.

[24] Zantema, H., and Geser, A. (1994), A complete characterization of termination of $0^p1^q \to 1^r0^s$, Report UU-CS-1994-44, Utrecht University. [To appear in Proceedings 6th Conference on Rewriting Techniques and Applications (RTA'95), Kaiserslautern, April 1995.]