BRICS

Basic Research in Computer Science

BRICS RS-96-22 Aceto & Fokkink: An Equational Axiomatization for Multi-Exit Iteration

An Equational Axiomatization for Multi-Exit Iteration

Luca Aceto
Wan J. Fokkink

BRICS Report Series **RS-96-22**

ISSN 0909-0878 **June 1996**

See back inner page for a list of recent publications in the BRICS Report Series. Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK - 8000 Aarhus C
Denmark

Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through WWW and anonymous FTP:

`http://www.brics.dk/`
`ftp ftp.brics.dk (cd pub/BRICS)`

An Equational Axiomatization for Multi-Exit Iteration

Luca Aceto*
BRICS†
Department of Computer Science
Aalborg University, Fr. Bajersvej 7E
9220 Aalborg Ø, Denmark

Wan Fokkink‡
Utrecht University, Department of Philosophy
Heidelberglaan 8, 3584 CS Utrecht
The Netherlands

Abstract

This paper presents an equational axiomatization of bisimulation equivalence over the language of Basic Process Algebra (BPA) with multi-exit iteration. Multi-exit iteration is a generalization of the standard binary Kleene star operation that allows for the specification of agents that, up to bisimulation equivalence, are solutions of systems of recursion equations of the form

$$\begin{array}{rcl} X_1 & = & P_1X_2 + Q_1 \\ & \vdots & \\ X_n & = & P_nX_1 + Q_n \end{array}$$

where n is a positive integer, and the P_i and the Q_i are process terms. The addition of multi-exit iteration to BPA yields a more expressive language than that obtained by augmenting BPA with the standard binary Kleene star (BPA*). As a consequence, the proof of completeness of the proposed equational axiomatization for this language, although standard in its general structure, is much more involved than that for BPA*. An expressiveness hierarchy for the family of k-exit iteration operators proposed by Bergstra, Bethke and Ponse is also offered.

AMS Subject Classification (1991): 08A70, 03C05, 68Q10, 68Q40, 68Q55, 68Q68, 68Q70.
CR Subject Classification (1991): D.3.1, F.1.2, F.3.2.
Keywords and Phrases: Concurrency, process algebra, Basic Process Algebra (BPA), multi-exit iteration, bisimulation, equational logic, complete axiomatizations.

*On leave from the School of Cognitive and Computing Sciences, University of Sussex, Brighton BN1 9QH, UK. Partially supported by the Human Capital and Mobility cooperation project Express. Email: `luca@iesd.auc.dk`.

†Basic Research in Computer Science, Centre of the Danish National Research Foundation.

‡Email: `fokkink@phil.ruu.nl`.

1 Introduction

One of the classic topics in the theory of computation is the study of axiomatic characterizations of equalities between variations on regular expressions. This field of research has been active since Kleene's original paper [27], where regular expressions were first introduced, and has yielded a collection of very deep and beautiful mathematical results. (The interested reader is invited to consult, e.g., [38, 14, 35, 29, 28] for an overview of the results that have been obtained within this line of research.) According to the point of view of formal language theory, a regular expression denotes a regular language, and two regular expressions are equal exactly when they denote the same language. This notion of semantics for regular expressions is the natural one to choose when the finite automaton associated with a regular event by Kleene's *synthesis theorem* (cf., e.g., [27, 38]) is viewed as accepting a language. However, as first observed by Milner [30], defining the semantics of an automaton as the language it accepts is inappropriate when one views it as a *reactive system*, i.e., as a system that computes by reacting to stimuli from its environment. For this reason, a wealth of notions of equivalence over processes that, to different degrees, capture their behaviour as reactive machines have been proposed in the literature on process theory. (The interested reader is invited to consult the references [22, 24] for more details.) Amongst these, the notion of bisimulation equivalence [34] has emerged as a fundamental semantic equivalence for reactive systems, and the development of its theory has by now reached a level of maturity that is comparable to that of the standard notions from the theory of formal languages. For example, the complete axiomatization of bisimulation equivalence for the regular fragment of CCS [32], provided by Milner in his classic paper [31], parallels those obtained by Salomaa for regular languages [37, 38], and have contributed to the realization that the notion of *process* is at least as elegant and mathematically tractable as that of *language*.

Despite these successes, process theory has traditionally lacked a systematic investigation of (equational) axiomatizations of process equivalences over regular expressions—a notable exception being Milner's seminal paper [31], where an implicational proof system was proposed for bisimulation equivalence over regular events, and the problem of its completeness raised. (To the best of the authors' knowledge, this problem of Milner's is still awaiting a solution.)

The study of axiomatic questions for variations on the language of regular expressions from the perspective of process theory has received new impulse after the publication of [9]. In *op. cit.* the authors have investigated the expressive power of variations on standard process description languages in which infinite behaviours are defined by means of Kleene's star operation [27, 15] rather than by means of systems of recursion equations, and have proposed an axiom system for bisimulation equivalence over the language of Basic Process Algebra [10] with the original binary version of the Kleene star operation (BPA*). The completeness of the axiom system proposed *ibidem* was proven by Fokkink and Zantema in [20, 17], and this result has been followed by a series of contributions in which several notions of process equivalence have been equationally axiomatized over languages incorporating variations on the Kleene star operation—see, e.g., [16, 3, 2, 1, 21, 18].

Interestingly, as already noted by Milner in [31, Sect. 6], not every process defined using finite-state systems of recursion equations can be described, up to bisimulation equivalence, using only regular expressions. (As shown in [9], any regular process can be specified in the axiom system ACP_τ [11] with Kleene star using handshake communication.) The limited expressive power of the Kleene star operation in denoting finite automata modulo bisimulation equivalence is highlighted in [8], where it is shown that the process described by the following recursion equation

$$X \stackrel{\text{def}}{=} a \cdot (a \cdot X + b) + a \qquad (1)$$

can*not* be expressed in the language BPA^* modulo bisimulation equivalence. (Of course, a simple use of Arden's lemma yields that the language denoted by X is the same associated with the regular expression $(a \cdot a)^*(a \cdot b + a)$. However, the process denoted by X above is *not* bisimulation equivalent to that associated with the BPA^* term $(a \cdot a)^*(a \cdot b + a)$.)

In order to increase the expressive power of super-languages of BPA incorporating Kleene star-like operations, the use of k-exit iteration has been proposed in [8]. For every positive integer k, and process terms P_i and Q_i $(1 \leq i \leq k)$, the process term $(P_1, \ldots, P_k)^*(Q_1, \ldots, Q_k)$ denotes a solution to the following list of recursion equations:

$$\begin{aligned} X_1 &= P_1 X_2 + Q_1 \\ &\vdots \\ X_k &= P_k X_1 + Q_k \end{aligned}$$

For example, the term $(a, a)^*(a, b)$ uses 2-exit iteration, and the reader will immediately realize that, up to isomorphism, it denotes the finite automaton associated with the variable X in (1).

The aim of this paper is to present an equational axiomatization of bisimulation equivalence over the language obtained by augmenting Basic Process Algebra with the family of k-exit iteration operations. In fact, we shall consider a slight syntactic generalization of the family of k-exit iteration operations, in that we permit the construction of terms of the form $(P_1, \ldots, P_m)^*(Q_1, \ldots, Q_n)$, with m and n two arbitrary positive integers. The result is a purely algebraic process language that is more expressive than BPA^*.

Apart from the standard laws from the equational theory for BPA, and the adaptations of the three axioms for the binary Kleene star [20] to multi-exit iteration, the equational axiomatization that we propose contains the following axiom schemas for multi-exit iteration:

$$\begin{array}{lrcl} \text{MEI4} & (x_0, x_1, \vec{v})^*(y, x_2(\vec{v}, x_0(x_1 + x_2))^*(\vec{w}, y), \vec{w}) & = & (x_0(x_1 + x_2), \vec{v})^*(y, \vec{w}) \\ \text{MEI5} & ((x, \vec{v})^m)^*((y, \vec{w})^n) & = & (x, \vec{v})^*(y, \vec{w}) \end{array}$$

where we use $\vec{v}$ and $\vec{w}$ to stand for arbitrary vectors of process variables, and n, m are positive integers. Typically, these two axioms deal with equalities between terms that have distinct exit degrees.

Our proof of the completeness of this axiom system with respect to bisimulation equivalence over BPA with multi-exit iteration uses an adaptation of a proof technique developed by the second author in [17]. The actual details of the completeness proof are, however, more delicate and intricate than those of the proof given *ibidem*. In particular, as the reader will realize, extra care need be taken in dealing with equalities whose equational deduction makes an essential use of the new laws MEI4–5.

As remarked above, Bergstra, Bethke and Ponse [8, 9] have shown that the process described by the recursion equation (1) cannot be specified in the language BPA* modulo bisimulation equivalence. In this paper, we generalize this result by showing that, in presence of a countably infinite set of actions, the family of k-exit iteration operations from [8] induces a strict expressiveness hierarchy of super-languages of BPA. More precisely, we prove that, for every positive integer k, the agent that corresponds to the solution of the following system of recursion equations (with X_1 as the leading variable)

$$\begin{array}{rcl} X_1 & = & a_1 X_2 + a_1 \\ & \vdots & \\ X_{k+1} & = & a_1 X_1 + a_{k+1} \end{array}$$

can only be specified, modulo bisimulation equivalence, with the use of multi-exit iterations of the form $(P_1, \ldots, P_m)^*(Q_1, \ldots, Q_n)$ with n greater than k.

We conclude this introduction with a brief overview of the contents of this paper. We begin by introducing the language of Basic Process Algebra with multi-exit iteration and its operational semantics (Sects. 2.1–2.3). The equational axiomatization of bisimulation equivalence over Basic Process Algebra with multi-exit iteration is presented in Sect. 2.4. The whole of Sect. 3 is devoted to a detailed proof of the completeness of the proposed axiomatization. The proof we present consists of three steps. First we isolate a collection of basic process terms, which cover, up to bisimulation equivalence, the whole language of process terms, and whose structure will simplify the proof of the promised completeness theorem (Sect. 3.1). We then proceed to define a well-founded ordering over basic terms which will allow for an inductive proof of our main result, and study some of its properties (Sect. 3.2). Finally, we shall show that two bisimilar basic terms can be proven equal using the equations in the proposed axiom system (Sect. 3.4). The paper concludes with an expressiveness hierarchy for the family of k-exit iteration operations (Sect. 4).

2 BPA with Multi-Exit Iteration

We begin by presenting the language of Basic Process Algebra with multi-exit iteration and its semantics.

2.1 The Syntax

We assume a non-empty alphabet A of atomic actions, with typical elements a, b, and a countably infinite set Var of process variables, disjoint from A, with typical elements x, y, z. We shall use α, β to range over $A \cup \mathsf{Var}$.

The language $\mathbb{T}(\mathrm{BPA}^{me*}(A))$ of terms over Basic Process Algebra (BPA) with multi-exit iteration is defined inductively as follows:

- each $\alpha \in A \cup \mathsf{Var}$ is a term;
- if P and Q are terms, then $P + Q$ is a term;
- if P and Q are terms, then $P \cdot Q$ is a term;
- if $P_1, ..., P_m$ and $Q_1, ..., Q_n$ $(m, n \geq 1)$ are terms, then $(P_1, ..., P_m)^*(Q_1, ..., Q_n)$ is a term.

The set of closed terms, i.e., terms that do not contain occurrences of process variables, is denoted by $\mathrm{T}(\mathrm{BPA}^{me*}(A))$. We shall use $P, Q, \ldots, Y$ to range over $\mathbb{T}(\mathrm{BPA}^{me*}(A))$. In writing terms over the above syntax, we shall always assume that the operation $\cdot$ binds stronger than $+$. In the sequel the operation $\cdot$ will often be omitted, so PQ denotes $P \cdot Q$. We shall use the symbol $\equiv$ to stand for syntactic equality of terms. The size of a process term is the number of operations occurring in it. A (closed) substitution is a mapping from process variables to (closed) terms in the language $\mathbb{T}(\mathrm{BPA}^{me*}(A))$. For every term P and (closed) substitution σ, the (closed) term obtained by replacing every occurrence of a variable x in P with the (closed) term $\sigma(x)$ will be written $P\sigma$.

Intuitively, closed terms stand for agents whose behaviour is completely specified, whereas terms containing occurrences of process variables denote agents with partially specified behaviour. For example, an atomic action a stands for a process that can only perform itself in one computational step and terminate in doing so; on the other hand, the term $a + x$ denotes a partially specified process, whose behaviour depends in part on that of the process term that is substituted for the variable x.

Apart from actions and variables, the signature of the language $\mathbb{T}(\mathrm{BPA}^{me*}(A))$ includes the binary operations of alternative composition $+$ and sequential composition $\cdot$ familiar from the theory of Basic Process Algebra [10, 7], and a variation on the original binary version of the Kleene star operation [27], that will be referred to as multi-exit iteration. For positive integers m and n, the process term $(P_1, ..., P_m)^*(Q_1, ..., Q_n)$ stands for an agent whose behaviour is specified by the following defining equation:

$$(P_1, ..., P_m)^*(Q_1, ..., Q_n) \;=\; P_1 \cdot (P_2, ..., P_m, P_1)^*(Q_2, ..., Q_n, Q_1) + Q_1 \;.$$

Multi-exit iteration is a mild syntactic generalization of the family of k-exit iteration operations introduced in [8]—the only difference being that in *op. cit.* the number of the P_i must always be equal to that of the Q_j. As we shall see, multi-exit iteration and the family of k-exit iteration operations have the same expressive power modulo bisimulation equivalence. (Cf. Sect. 4 for some remarks on the expressive power of the k-exit iteration operations.)

In order to simplify notation in the presentation of the operational semantics and of the axiomatization for $\mathbb{T}(\mathrm{BPA}^{me*}(A))$, we shall use the notion of 'vectors of processes'. A vector of processes is a tuple $(P_1, ..., P_m)$, where $m \geq 0$. We shall use $\vec{P}, \vec{Q}, \vec{R}, \vec{S}$ to denote such vectors of processes. In multi-exit iteration, the expressions at the left- and right-hand side of the star are non-empty vectors of processes.

In the sequel, $(Q, \vec{P})$ represents the vector that is obtained by concatenating the process term Q in front of vector $\vec{P}$, and $(\vec{P}, Q)$ represents the vector that is obtained by appending the process term Q at the rear of vector $\vec{P}$. Furthermore, we assume the following features for vectors:

- Multiplication with a process term: $(P_1, ..., P_m) \cdot Q$ equals $(P_1 Q, ..., P_m Q)$.
- Power of a vector: for a positive integer n, $(P_1, ..., P_m)^n$ equals

$$(\underbrace{P_1, ..., P_m, \ldots, P_1, ..., P_m}_{n \text{ times}})$$

Enclosing parentheses will always be omitted from vectors of length one, i.e., (P) will be written P.

2.2 Operational Semantics

The operational semantics for the language $\mathbb{T}(\mathrm{BPA}^{me*}(A))$ is given by the labelled transition system [26, 36]

$$\left(\mathbb{T}(\mathrm{BPA}^{me*}(A)), \left\{\xrightarrow{\alpha} \mid \alpha \in A \cup \mathsf{Var}\right\}, \left\{\xrightarrow{\alpha}\surd \mid \alpha \in A \cup \mathsf{Var}\right\}\right)$$

where the transition relations $\xrightarrow{\alpha}$ and the unary predicates $\xrightarrow{\alpha}\surd$ are, respectively, the least subsets of $\mathbb{T}(\mathrm{BPA}^{me*}(A)) \times \mathbb{T}(\mathrm{BPA}^{me*}(A))$ and $\mathbb{T}(\mathrm{BPA}^{me*}(A))$ satisfying the rules in Table 1. Intuitively, a transition $P \xrightarrow{a} Q$ means that the system represented by the term P can perform the action a, thereby evolving into Q, whereas $P \xrightarrow{x} P'$ means that the initial behaviour of P may depend on the term that is substituted for the process variable x. The special symbol $\surd$ stands for (successful) termination; therefore the interpretation of the statement $P \xrightarrow{\alpha}\surd$ is that the process term P can terminate by performing α, if α is an atomic action, or by executing to completion the term that is substituted for the process variable x, if $\alpha = x$.

$$\frac{}{\alpha \xrightarrow{\alpha}\surd}$$

$$\frac{P \xrightarrow{\alpha}\surd}{P + Q \xrightarrow{\alpha}\surd} \qquad \frac{Q \xrightarrow{\alpha}\surd}{P + Q \xrightarrow{\alpha}\surd} \qquad \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$$

$$\frac{P \xrightarrow{\alpha}\surd}{P \cdot Q \xrightarrow{\alpha} Q} \qquad \frac{P \xrightarrow{\alpha} P'}{P \cdot Q \xrightarrow{\alpha} P' \cdot Q}$$

$$\frac{P \xrightarrow{\alpha}\surd}{(P, \vec{Q})^*(R, \vec{S}) \xrightarrow{\alpha} (\vec{Q}, P)^*(\vec{S}, R)} \qquad \frac{P \xrightarrow{\alpha} P'}{(P, \vec{Q})^*(R, \vec{S}) \xrightarrow{\alpha} P' \cdot (\vec{Q}, P)^*(\vec{S}, R)}$$

$$\frac{R \xrightarrow{\alpha}\surd}{(P, \vec{Q})^*(R, \vec{S}) \xrightarrow{\alpha}\surd} \qquad \frac{R \xrightarrow{\alpha} R'}{(P, \vec{Q})^*(R, \vec{S}) \xrightarrow{\alpha} R'}$$

Table 1: Transition Rules

Definition 2.1 *The term P' is a* derivative *of P if P can evolve into P' by zero or more transitions. A derivative P' of P is* proper *if P can evolve into P' by performing at least one transition.*

The following basic fact can be easily shown by structural induction on terms:

Fact 2.2 *For every $P \in \mathbb{T}(\mathrm{BPA}^{me*}(A))$, the set of derivatives of P is finite.*

Process terms are considered modulo bisimulation equivalence from Park [34]. Intuitively, two process terms are bisimilar if they have the same branching structure.

Definition 2.3 *Two process terms P and Q are* bisimilar*, denoted by $P \leftrightarrow Q$, if there exists a symmetric binary relation $\mathcal{B}$ on process terms which relates P and Q, such that:*

- *if $R \,\mathcal{B}\, S$ and $R \xrightarrow{\alpha} R'$, then there is a transition $S \xrightarrow{\alpha} S'$ such that $R' \,\mathcal{B}\, S'$,*
- *if $R \,\mathcal{B}\, S$ and $R \xrightarrow{\alpha} \surd$, then $S \xrightarrow{\alpha} \surd$.*

Such a relation $\mathcal{B}$ will be called a bisimulation *(witnessing the equivalence $P \leftrightarrow Q$). The relation $\leftrightarrow$ will be referred to as* bisimulation equivalence*.*

Note that if P is bisimilar to Q, then every (proper) derivative of P is bisimilar to some (proper) derivative of Q, and vice versa.

The transition rules in Table 1 are in the 'path' format of Baeten and Verhoef [6]. Hence, bisimulation equivalence is a congruence with respect to all the operations in the signature of $\mathbb{T}(\mathrm{BPA}^{me*}(A))$. The interested reader is invited to consult [6] for the definition of the path format for operational rules, and for a proof of the aforementioned congruence result.

Remark: The proof of the congruence theorem for the path format presented in *op. cit.* uses the extra assumption that the rules are well-founded. Fokkink and van Glabbeek [19] have shown that this requirement, which is, however, met by the rules in Table 1, can be dropped.

The reader might have noticed that we have defined notions of operational semantics and bisimulation equivalence that apply to open terms directly, following [33, 23] for process algebra with abstraction, and [1] for process algebra with the prefix iteration operation from [16], which is a restricted version of the Kleene star. This approach deviates from the standard practice in process theory, which prescribes to define operational semantics and bisimulation equivalence for closed terms only, and to give meaning to open terms thus:

$$P \leftrightarrow Q \;\triangleq\; P\sigma \leftrightarrow Q\sigma \text{ for all closed substitutions } \sigma : \mathsf{Var} \to \mathrm{T}(\mathrm{BPA}^{me*}(A)).$$

(Note that this amounts to stipulating that two open terms are equivalent exactly when the equation $P = Q$ holds in the algebra of closed terms modulo bisimulation equivalence.) The following result shows that both approaches yield the same notion of bisimulation equivalence over $\mathbb{T}(\mathrm{BPA}^{me*}(A))$, that is, in our setting, two open terms are bisimilar if and only if all their closed instantiations are.

Lemma 2.4 *For every $P, Q \in \mathbb{T}(\mathrm{BPA}^{me*}(A))$,*

$$P \leftrightarrow Q \quad \Leftrightarrow \quad P\sigma \leftrightarrow Q\sigma \textit{ for all closed substitutions } \sigma : \mathsf{Var} \to \mathrm{T}(\mathrm{BPA}^{me*}(A)).$$

Proof: The result can be proven following the strategy that was employed in [1] for prefix iteration. We remark, however, that the technical details are considerably more complicated in the case of multi-exit iteration. In particular, in the proof of the 'if implication', the choice of a canonical closed substitution $\sigma_{(P,Q)}$ with the property that

$$P\sigma_{(P,Q)} \leftrightarrow Q\sigma_{(P,Q)} \quad \Rightarrow \quad P \leftrightarrow Q$$

requires more care than in the aforementioned reference. The interested reader is invited to consult [39, Theorems 3 and 4], where this type of result is proven in the more general setting of a simply typed lambda calculus, which captures multi-exit iteration. $\Box$

Lemma 2.4 implies that any complete axiomatization of bisimulation equivalence over the language $\mathbb{T}(\mathrm{BPA}^{me*}(A))$ is also a complete axiomatization of the algebra of closed terms modulo bisimulation. Such an axiomatization will, *a fortiori*, be ω-complete in the sense of, e.g., [25].

2.3 Norm and L-value of Process Terms

Process terms in $\mathbb{T}(\mathrm{BPA}^{me*}(A))$ are *normed*, which means that they are able to terminate by embarking in a finite sequence of transitions. We call such a sequence a termination trace. The *norm* of a process term P, denoted by $|P|$, is the length of its shortest termination trace; this notion stems from [4]. Note that bisimilar process terms have the same norm. The following lemma, which is due to Caucal [13], is typical for normed processes. The interested reader is referred to [17] for its proof.

Lemma 2.5 *Let $P, Q, R, S \in \mathbb{T}(\mathrm{BPA}^{me*}(A))$ be such that $PQ \leftrightarrow RS$. Then the following statements hold:*

- *If $|Q| = |S|$, then $P \leftrightarrow R$ and $Q \leftrightarrow S$.*
- *If $|Q| < |S|$, then there is a proper derivative P' of P such that $P \leftrightarrow RP'$ and $P'Q \leftrightarrow S$.*

The notion of norm may be used as a measure of the complexity of terms which is useful in inductive proofs. (Cf., e.g., the proof of the above lemma in [17].) However, this measure of term complexity does not lend itself to use in completeness proofs like the one presented in *op. cit.* because it does not respect term size. For example, the term $aa + a$ has smaller norm than its sub-term aa. For this reason, Fokkink and Zantema [20] introduced the notion of *L-value*, which does not have this drawback.

Definition 2.6 *The L-value of a process term P, notation $L(P)$, is defined thus:*

$$L(P) \quad \triangleq \quad \sup\{|P'| \mid P' \text{ a proper derivative of } P\} \ .$$

Note that, as the set of derivatives of P is finite for every term $P \in \mathbb{T}(\mathrm{BPA}^{me*}(A))$ (Fact 2.2), the L-value of a process term is a well-defined natural number. For example, $L(\alpha) = \sup \varnothing = 0$.

The following basic properties of the notion of L-value will be useful in the technical developments to follow.

Lemma 2.7

1. *If $P \leftrightarrow Q$, then $L(P) = L(Q)$;*

2. *$L(P) < L(PQ)$;*

3. *$L(P_i) < L\Big((P_1, ..., P_m)^*(Q_1, ..., Q_n)\Big)$ for $i = 1, ..., m$.*

Remark: Note that, in general, the L-values of Q and Q_i are *not* smaller than those of PQ and $(P_1, ..., P_m)^*(Q_1, ..., Q_n)$, respectively. For instance, if $P \equiv a$ and $Q \equiv a + aa$, then $L(PQ) = L(Q) = 1$. The reader will find the construction of a similar example for multi-exit iteration an easy exercise.

2.4 The Axiom System

Table 2 contains our axiom system $\mathcal{E}$ for $\mathbb{T}(\mathrm{BPA}^{me*}(A))$ modulo bisimulation equivalence. It consists of the standard axioms A1–5 from Basic Process Algebra (cf. [10, 7]) together with five axiom schemas MEI1–5 for multi-exit iteration. In these axiom schemas, $\vec{v}$ and $\vec{w}$ denote meta-variables which range over vectors of processes, and n, m range over the set of positive integers.

Remark: For the sake of clarity, we remark that a meta-variable $\vec{v}$ is just syntactic sugar for an arbitrary vector of process variables. Therefore each of the axiom schemas MEI1–5 stands for an infinite family of equations, viz. one for each instantion of the meta-variables ranging over vectors of process variables and of those ranging over positive integers.

Laws MEI1–3 are modifications of the standard axioms for the binary Kleene star which, together with A1–5, have been shown to be complete for bisimulation equivalence over the language BPA^* in [20, 17]. In particular, axiom MEI3 is the multi-exit version of a law which originates from Troeger's work [40]. Axioms MEI4 and MEI5 are, to the best of our knowledge, new.

Proposition 2.8 (Soundness) *If $\mathcal{E} \vdash P = Q$, then $P \leftrightarrow Q$.*

Proof: Since bisimulation equivalence is a congruence, this can be verified by checking soundness for each axiom separately, which is left to the reader. $\Box$

Remark: The fact that multi-exit iteration and the family of k-exit iteration operations ($k \geq 1$), as defined in [8], have the same expressive power modulo bisimulation equivalence is witnessed by the soundness of axiom MEI5. This axiom may be used to turn every instance of a multi-exit iteration into an equivalent one that uses k-exit iteration for an appropriate k.

In the proof of the completeness of the axiom system $\mathcal{E}$ for bisimulation equivalence over the language $\mathbb{T}(\mathrm{BPA}^{me*}(A))$, we shall find it useful to have more general variants of the axioms MEI3,4.

Let us assume that $\vec{v}_0$ and $\vec{w}_0$ are vectors of process variables of the *same* length. The following equation, which we shall refer to as MEI3′, can be derived easily from MEI3 using MEI1:

$$(\vec{v}_0, x_1, \vec{v}_1)^*(\vec{w}_0, y + x_2(\vec{v}_1, \vec{v}_0, x_1 + x_2)^*(\vec{w}_1, \vec{w}_0, y), \vec{w}_1) = (\vec{v}_0, x_1 + x_2, \vec{v}_1)^*(\vec{w}_0, y, \vec{w}_1) \ .$$

A1	$x+y$	$=$	$y+x$
A2	$(x+y)+z$	$=$	$x+(y+z)$
A3	$x+x$	$=$	x
A4	$(x+y)z$	$=$	$xz+yz$
A5	$(xy)z$	$=$	$x(yz)$
MEI1	$x(\vec{v},x)^*(\vec{w},y)+y$	$=$	$(x,\vec{v})^*(y,\vec{w})$
MEI2	$(x,\vec{v})^*(y,\vec{w})z$	$=$	$(x,\vec{v})^*(yz,\vec{w}z)$
MEI3	$(x_1,\vec{v})^*(y+x_2(\vec{v},x_1+x_2)^*(\vec{w},y),\vec{w})$	$=$	$(x_1+x_2,\vec{v})^*(y,\vec{w})$
MEI4	$(x_0,x_1,\vec{v})^*(y,x_2(\vec{v},x_0(x_1+x_2))^*(\vec{w},y),\vec{w})$	$=$	$(x_0(x_1+x_2),\vec{v})^*(y,\vec{w})$
MEI5	$((x,\vec{v})^m)^*((y,\vec{w})^n)$	$=$	$(x,\vec{v})^*(y,\vec{w})$

Table 2: The Axiom System $\mathcal{E}$

Similarly, the following equation, which we shall refer to as MEI4′, can be derived easily from MEI4 using MEI1:

$$(\vec{v}_0,x_0,x_1,\vec{v}_1)^*(\vec{w}_0,y,x_2(\vec{v}_1,\vec{v}_0,x_0(x_1+x_2))^*(\vec{w}_1,\vec{w}_0,y),\vec{w}_1) = (\vec{v}_0,x_0(x_1+x_2),\vec{v}_1)^*(\vec{w}_0,y,\vec{w}_1)\ .$$

Note that the soundness of these generalized versions of equations MEI3–4 depends crucially on the assumption that $\vec{v}_0$ and $\vec{w}_0$ are vectors of process variables of the same length, and their use in the proof of the completeness theorem will be restricted to situations in which this requirement is met.

Notation 2.9 *For an axiom system $\mathcal{T}$, we write $\mathcal{T} \vdash P = Q$ iff the equation $P = Q$ is provable from the axioms in $\mathcal{T}$ using the rules of equational logic. For a collection of equations X over the signature of $\mathbb{T}(\mathrm{BPA}^{me*}(A))$, we write $P \stackrel{X}{=} Q$ as a short-hand for A1,A2,$X \vdash P = Q$.*

For $I = \{i_1,\ldots,i_n\}$ a finite, non-empty *index set, we write $\sum_{i\in I} P_i$ for $P_{i_1}+\cdots+P_{i_n}$.*

For notational convenience, and in order to reduce the number of cases in the completeness proof, in what follows we shall identify a term P with the meta-notations $\sum_{i\in\varnothing} Q_i + P$ and $(\sum_{i\in\varnothing} Q_i)Q + P$.

The collection of possible transitions of each process term P is non-empty and finite, say $\{P \xrightarrow{\alpha_i} P_i \mid i = 1,...,m\} \cup \{P \xrightarrow{\beta_j} \checkmark \mid j = 1,...,n\}$. We call the term

$$\sum_{i=1}^{m} \alpha_i P_i + \sum_{j=1}^{n} \beta_j$$

the *expansion* of P. The terms $\alpha_i P_i$ and β_j will be referred to as the *summands* of P.

Lemma 2.10 *Each process term is provably equal to its expansion.*

Proof: Straightforward, by structural induction on terms, using axioms A4, A5 and MEI1. □

3 The Completeness Proof

The remainder of the paper is devoted to a proof of completeness of the axiom system $\mathcal{E}$ with respect to bisimulation equivalence over the language $\mathbb{T}(\mathrm{BPA}^{me*}(A))$. The proof of this result will be given by adapting a proof technique developed by the second author in [17]. A comparison between this proof technique and the one originally used in [20] to solve the completeness problem for bisimulation equivalence over Basic Process Algebra with binary Kleene star may be found in [17].

The proof we present consists of three steps. First we isolate a collection of basic process terms, which cover, up to bisimulation equivalence, the whole language of process terms, and whose structure will simplify the proof of the promised completeness theorem. We then proceed to define a well-founded ordering over basic terms which will allow for an inductive proof of our main result, and study some of its properties. Finally, we shall show that two bisimilar basic terms can be proven equal using the equations in $\mathcal{E}$. This proof strategy will be familiar to readers acquainted with the literature on completeness results from process theory. The actual details of the proof that we now proceed to present are, however, rather challenging and novel.

3.1 Basic Terms

As a first stepping stone towards our main result, we now aim at constructing a collection of *basic* process terms, with the property that each process term is provably equal to a basic one. We shall then prove the completeness theorem by showing that bisimilar basic terms are provably equal.

Let $\mathbb{G}$ denote the collection of process terms in $\mathbb{T}(\mathrm{BPA}^{me*}(A))$ which only use action prefixing, in the sense of Milner [32], in lieu of general sequential composition, and in which vectors at the left- and right-hand side of multi-exit iteration have the same length. That is, $\mathbb{G}$ is defined inductively as follows:

- each $\alpha \in A \cup \mathsf{Var}$ is in $\mathbb{G}$;
- if P and Q are in $\mathbb{G}$, then $P + Q$ is in $\mathbb{G}$;
- if $\alpha \in A \cup \mathsf{Var}$ and P is in $\mathbb{G}$, then αP is in $\mathbb{G}$;
- if $P_1, ..., P_n$ and $Q_1, ..., Q_n$ $(n \geq 1)$ are in $\mathbb{G}$, then $(P_1, ..., P_n)^*(Q_1, ..., Q_n)$ is in $\mathbb{G}$.

Lemma 3.1 *Every process term can be proven equal to a term in* $\mathbb{G}$ *using axioms A4,5 and MEI2,5.*

The import of the above lemma is that, without loss of generality, we may restrict ourselves to considering terms in $\mathbb{G}$. However, $\mathbb{G}$ is not yet our desired set of basic terms. In fact, one of the most fundamental properties that we require of a collection of basic terms for use in the proof of the completeness theorem is that it be closed under transitions. This is a property that $\mathbb{G}$ does *not* enjoy. For example, the term $(a^*a)^*a$ is clearly in $\mathbb{G}$, and

$$(a^*a)^*a \xrightarrow{a} (a^*a)\Big((a^*a)^*a\Big) \ .$$

However, the right-hand side of the above transition is a term that is not contained in $\mathbb{G}$. In order to overcome this complication, we introduce the following collection $\mathbb{H}$ of process terms:

- if a term $(P_1, ..., P_n)^*(Q_1, ..., Q_n)$ is in $\mathbb{G}$, then it is also in $\mathbb{H}$;
- if a term $(P_1, ..., P_n)^*(Q_1, ..., Q_n)$ is in $\mathbb{G}$, and if P' is a proper derivative of P_n, then $P'(P_1, ..., P_n)^*(Q_1, ..., Q_n)$ is in $\mathbb{H}$.

For example, the term $(a^*a)\Big((a^*a)^*a\Big)$ is included in $\mathbb{H}$ because a^*a is a proper derivative of itself.

The set $\mathbb{B}$ of *basic* terms is the union of $\mathbb{G}$ and $\mathbb{H}$.

Lemma 3.2 *If $P \in \mathbb{B}$ and $P \xrightarrow{\alpha} P'$, then $P' \in \mathbb{B}$.*

Proof: Let $P \xrightarrow{\alpha} P'$. If $P \in \mathbb{H}\backslash\mathbb{G}$, then it is easy to see that $P' \in \mathbb{H}$. If $P \in \mathbb{G}$, then a straightforward structural induction yields that $P' \in \mathbb{B}$. □

For later use, we now define an equivalence relation $\cong$ on $\mathbb{H}$ as the least one satisfying axiom MEI5 and the equivalences

- $(P_1, ..., P_n)^*(Q_1, ..., Q_n) \ \cong\ (P_2, ..., P_n, P_1)^*(Q_2, ..., Q_n, Q_1)$;
- $(P_n, P_1, ..., P_{n-1})^*(Q_n, Q_1, ..., Q_{n-1}) \ \cong\ P'(P_1, ..., P_n)^*(Q_1, ..., Q_n)$ if P' is a proper derivative of P_n.

The key properties of the relation $\cong$ needed in the technical developments to follow will be studied in the following section.

3.2 An Ordering on Terms

Having identified a set of basic terms closed under transitions, we now proceed to define a well-founded ordering on this set that will allow us to give an inductive proof of the main result of this paper.

Definition 3.3 *We say that a term P is an* exit derivative *of a term Q iff P is a derivative of Q, but not vice versa.*

Let $\prec$ denote the least transitive relation over the set of process terms satisfying:

- $P \prec Q$ if $L(P) < L(Q)$,
- $P \prec Q$ if P is an exit derivative of Q.

Intuitively, if $P \prec Q$, then either the set of derivatives of P is properly included in that of Q, or the L-value of P is strictly smaller than that of Q.

Lemma 3.4 *$\prec$ is a well-founded ordering on $\mathbb{T}(\mathrm{BPA}^{me*}(A))$.*

Proof: First of all, observe that if $P \prec Q$ then $L(P) \leq L(Q)$. This follows because if P is a derivative of Q, then $L(P) \leq L(Q)$ since all proper derivatives of P are also proper derivatives of Q.

Suppose now, towards a contradiction, that $\prec$ is not well-founded. This means that there exists an infinite descending chain $P_0 \succ P_1 \succ P_2 \succ \cdots$. As $L(P_n) \geq L(P_{n+1})$ for all n, there is an N such that $L(P_N) = L(P_n)$ for all $n > N$. Since $P_m \succ P_n$ for $m, n > N$ with $m < n$, it follows that P_n is an exit derivative of P_m for each such m, n. By Fact 2.2, each process term has only finitely many derivatives, so there are $m, n > N$ with $m < n$ and $P_m \equiv P_n$. Then $P_m \not\succ P_n$, and we have found a contradiction. Hence, $\prec$ is well-founded. $\square$

Following [17], we now proceed to study the interaction between the operational semantics of process terms and the above-defined ordering. A technical tool we shall use below is a weight function g that associates a natural number to each process term. This is defined thus:

$$\begin{array}{rcl} g(\alpha) & \triangleq & 0 \\ g(P+Q) & \triangleq & \max\{g(P), g(Q)\} + 1 \\ g(PQ) & \triangleq & \max\{g(P), g(Q)\} \\ g\Big((P_1, ..., P_m)^*(Q_1, ..., Q_n)\Big) & \triangleq & \max\{g(P_1), ..., g(P_m), g(Q_1) + 1, ..., g(Q_n) + 1\} \ . \end{array}$$

The basic property of this weight function that we shall need is expressed in the lemma below, which follows by a straightforward structural induction.

Lemma 3.5 *If P' is a derivative of P, then $g(P') \leq g(P)$. Moreover, if*

- *$P \equiv P_1 + P_2$ for some terms P_1 and P_2, and P' is a proper derivative of P, or*
- *$P \equiv (P_1, ..., P_m)^*(Q_1, ..., Q_n)$, for some terms P_i $(1 \leq i \leq m)$ and Q_j $(1 \leq j \leq n)$, and P' is a proper derivative of some Q_j,*

then $g(P') < g(P)$.

The following two lemmas will play a major rôle in the proof of the main result of this paper. The first states that, intuitively, escaping from a loop reduces the complexity of a process term, as measured by the above defined ordering. This is due to the fact that once a process term has exited a loop, it can never return to it.

Lemma 3.6 *If Q' is a proper derivative of Q_i for some $i = 1, ..., n$, then*

$$Q' \prec (P_1, ..., P_m)^*(Q_1, ..., Q_n) \ .$$

Proof: If Q' is a proper derivative of some Q_i, then Lem. 3.5 gives that

$$g(Q') < g\big((P_1, ..., P_m)^*(Q_1, ..., Q_n)\big) \ .$$

Again using Lem. 3.5, we infer from this inequality that $(P_1, ..., P_m)^*(Q_1, ..., Q_n)$ is not a derivative of Q'. Since Q' is a derivative of $(P_1, ..., P_m)^*(Q_1, ..., Q_n)$, it follows that $Q' \prec (P_1, ..., P_m)^*(Q_1, ..., Q_n)$. $\square$

As witnessed by the proof of the above lemma, the exit derivatives of a term of the form $(P_1, ..., P_m)^*(Q_1, ..., Q_n)$ are exactly the proper derivatives of the terms Q_i. As an immediate consequence of this observation, we note that:

Fact 3.7 *If $P, Q \in \mathbb{H}$ and $P \cong Q$, then P and Q have the same L-value and exit derivatives.*

The observations collected in the above result imply that $\cong$-equivalent basic terms dominate the same process terms with respect to the ordering $\prec$, i.e., that if $P \prec R \cong S$, then $P \prec S$.

The following lemma is part of the crux of the proof of the completeness theorem; intuitively, it states that the only transitions between basic terms that do not decrease the complexity of terms, as measured by $\prec$, are those belonging to some loop.

Lemma 3.8 *If $P \in \mathbb{B}$ and $P \xrightarrow{\alpha} P'$, then either $P' \prec P$, or $P, P' \in \mathbb{H}$ and $P \cong P'$.*

Proof: We begin by establishing two facts that we shall use in the proof of the statement of the lemma.

A. If $P \in \mathbb{B}$, $P' \notin \mathbb{H}$ and $P \xrightarrow{\alpha} P'$, then P' has smaller size than P.

   Proof. First of all, note that the claim is vacuously true if $P \in \mathbb{H}\backslash\mathbb{G}$, because, in that case, it follows that $P' \in \mathbb{H}$. That the claim holds for $P \in \mathbb{G}$ can be shown by a simple structural induction on P.

B. If $P \in \mathbb{H}$ and $P \xrightarrow{\alpha} P'$, then either $g(P) > g(P')$, or $P' \in \mathbb{H}$ and $P \cong P'$.

   Proof. Since $P \in \mathbb{H}$, it follows that, for some terms P_i and Q_i $(1 \leq i \leq n)$,

   - either $P \equiv (P_1, ..., P_n)^*(Q_1, ..., Q_n)$,
   - or $P \equiv P'_n(P_1, ..., P_n)^*(Q_1, ..., Q_n)$, for some proper derivative P'_n of P_n.

   Hence, P' can have one of the following three forms:

   1. $P' \equiv (P_2, ..., P_n, P_1)^*(Q_2, ..., Q_n, Q_1)$,
   2. $P' \equiv P''_n(P_1, ..., P_n)^*(Q_1, ..., Q_n)$, for some proper derivative P''_n of P_n, or
   3. $P' \equiv Q'_1$, for some proper derivative of Q_1.

   In the first two cases $P' \in \mathbb{H}$ and $P \cong P'$, and in the last case $g(P') = g(Q'_1) < g(P)$ (Lem. 3.5).

We are now in a position to prove the lemma. Assume that, for some basic term P, $P \xrightarrow{\alpha} P'$ and $P' \not\prec P$. We prove that $P, P' \in \mathbb{H}$ and $P \cong P'$.

To this end, note, first of all, that, since P' is a proper derivative of P and $P' \not\prec P$, it must be the case that P is a proper derivative of P'. So there exists a sequence of transitions

$$P_0 \xrightarrow{\alpha_1} P_1 \xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_n} P_n, \qquad n \geq 2,$$

where $P_0 \equiv P \equiv P_n$ and $P_1 \equiv P'$. Note that each term P_k, $0 \leq k \leq n$, is basic as P is (Lem. 3.2).

We claim that $P_l \in \mathbb{H}$ for some $0 \leq l \leq n$. In fact, assume, towards a contradiction, that $P_k \notin \mathbb{H}$ for all k. Then fact A implies that P_{k+1} has smaller size than P_k for $k = 0, ..., n-1$. Therefore P has smaller size than itself, which is impossible.

Hence, $P_l \in \mathbb{H}$ for some l. Since each P_k is a proper derivative of each $P_{k'}$, we have $g(P_k) \leq g(P_{k'})$ for all k and k' (Lem. 3.5). Therefore $g(P_k)$ must be the same for all k. Now fact B and $P_l \in \mathbb{H}$ imply that $P_k \in \mathbb{H}$ for all k, and $P_0 \cong P_1 \cong \cdots \cong P_n$. □

Elements of $\mathbb{B} \times \mathbb{B}$ are considered modulo commutativity. The well-founded ordering $\prec$ on $\mathbb{B}$ induces an ordering on $\mathbb{B} \times \mathbb{B}$ as the least transitive relation satisfying:

$$(P, Q) \sqsubset (R, S) \quad \text{if} \quad P \prec R \text{ and } Q \cong S \ .$$

It is immediate to see that the above-defined ordering on $\mathbb{B} \times \mathbb{B}$ is also well-founded.

3.3 A Lemma

In the proof of the completeness theorem to follow, we shall often need to analyze equivalences between terms of the form $P'(P_1, ..., P_m)^*(Q_1, ..., Q_m)$. We now aim at establishing a lemma that will be a useful tool in the study of these equivalences.

Notation 3.9 *For a term* $X \equiv (P_1, ..., P_n)^*(Q_1, ..., Q_n)$*, the expression* $\mathrm{shift}(X)$ *stands for the term* $(P_2, ..., P_n, P_1)^*(Q_2, ..., Q_n, Q_1)$*. For a non-negative integer* k*,* $\mathrm{shift}^k(X)$ *denotes the result of applying the function* shift *to the term* X k *times. Note that* $X \underline{\leftrightarrow} P_1\mathrm{shift}(X) + Q_1$*.*

The following lemma will be applied in the proof of Lem. 3.11.

Lemma 3.10 *Let* $X \equiv (P_1, ..., P_m)^*(Q_1, ..., Q_m)$ *and* $Y \equiv (R_1, ..., R_n)^*(S_1, ..., S_n)$*. Assume that* $TX \underline{\leftrightarrow} UY$ *for some* T *and* U*, and that* Q_1 *does not have a proper derivative that is bisimilar to* Y*. Then we have the following two possibilities:*

1. *either* $T \underline{\leftrightarrow} U$*;*
2. *or* $T \underline{\leftrightarrow} U(V + W)$ *for some terms* V *and* W*, whose proper derivatives are contained in those of* T*, such that* $VX \underline{\leftrightarrow} R_1\mathrm{shift}(Y)$ *and* $WX \underline{\leftrightarrow} S_1$*.*

Proof: Lem. 2.5 applied to the equivalence $TX \underline{\leftrightarrow} UY$ yields three possibilities:

A. either $T \underline{\leftrightarrow} U$ and $X \underline{\leftrightarrow} Y$;

B. or $T \underline{\leftrightarrow} UT'$ and $T'X \underline{\leftrightarrow} Y$ for some proper derivative T' of T;

C. or $TU' \underline{\leftrightarrow} Y$ and $X \underline{\leftrightarrow} U'Y$ for some proper derivative U' of U.

Case A agrees with the first statement in the lemma.

If case B holds, then $T'X \underline{\leftrightarrow} Y \underline{\leftrightarrow} R_1\mathrm{shift}(Y) + S_1$. Then clearly there exist terms V and W, whose proper derivatives are contained in those of T', such that $T' \underline{\leftrightarrow} V+W$, $VX \underline{\leftrightarrow} R_1\mathrm{shift}(Y)$, and $WX \underline{\leftrightarrow} S_1$. This agrees with the second statement in the lemma.

Finally, case C contradicts one of the assumptions. In fact, as $X \underline{\leftrightarrow} U'Y$ implies $P_1\mathrm{shift}(X) + Q_1 \underline{\leftrightarrow} U'Y$, in that case Q_1 has a proper derivative that is bisimilar to Y. □

The following technical lemma will be used repeatedly in the proof of the promised completeness theorem.

Lemma 3.11 *Let* $X \equiv (P_1, ..., P_m)^*(Q_1, ..., Q_m)$ *and* $Y \equiv (R_1, ..., R_n)^*(S_1, ..., S_n)$*. Assume that* $T_1X \underline{\leftrightarrow} R_1\mathrm{shift}(Y)$ *for some term* T_1*, and that* Q_1 *has no proper derivative that is bisimilar to* Y*. Then we have the following two possibilities:*

I. *either there are terms* $U_1, ..., U_n$*, whose proper derivatives are contained in those of* T_1*, such that:*

$$\begin{array}{rcll} T_1 & \underline{\leftrightarrow} & R_1(R_2, ..., R_n, R_1)^*(U_2, ..., U_n, U_1) & \\ U_iX & \underline{\leftrightarrow} & S_i & i = 1, ..., n \end{array}$$

II. or there is a $k \in \{1, ..., n\}$ and there are terms $T_2, ..., T_k$ and $U_2, ..., U_k$, whose proper derivatives are contained in those of T_1, such that:

$$\begin{array}{rcll} T_i & \underline{\leftrightarrow} & R_i(T_{i+1} + U_{i+1}) & i = 1, ..., k-1 \\ T_k & \underline{\leftrightarrow} & R_k & \\ T_i X & \underline{\leftrightarrow} & R_i \mathrm{shift}^i(Y) & i = 2, ..., k \\ U_i X & \underline{\leftrightarrow} & S_i & i = 2, ..., k \ . \end{array}$$

Proof: Assume that the proviso of the lemma holds for some terms X, Y and T_1 of the required form. Consider the following equivalence A_i for some $i \leq n$:

A_i $T_i X \underline{\leftrightarrow} R_i \mathrm{shift}^i(Y)$.

Recall that we assumed that it holds for $i = 1$.

As Q_1 has no proper derivative that is bisimilar to Y, and Y is a proper derivative of $\mathrm{shift}^i(Y)$ for every non-negative integer i, it follows that Q_1 has no proper derivative that is bisimilar to $\mathrm{shift}^i(Y)$. Therefore we can apply Lem. 3.10 to equivalence A_i to obtain the following possibilities; either

B_i $T_i \underline{\leftrightarrow} R_i$,

or, if $i < n$, then there exist terms T_{i+1} and U_{i+1}, whose proper derivatives are contained in those of T_i, such that

C_i $T_i \underline{\leftrightarrow} R_i(T_{i+1} + U_{i+1})$, $T_{i+1} X \underline{\leftrightarrow} R_{i+1} \mathrm{shift}^{i+1}(Y)$ and $U_{i+1} X \underline{\leftrightarrow} S_{i+1}$,

or, if $i = n$, then there exist terms T_{n+1} and U_1, whose proper derivatives are contained in those of T_n, such that

C_n $T_n \underline{\leftrightarrow} R_n(T_{n+1} + U_1)$, $T_{n+1} X \underline{\leftrightarrow} R_1 \mathrm{shift}(Y)$ and $U_1 X \underline{\leftrightarrow} S_1$.

Note that, by transitivity, the proper derivatives of the T_i and the U_i are all contained in those of T_1.

In light of the above discussion, Lem. 3.10 gives that condition A_i implies either B_i or C_i for $i = 1, ..., n$. Furthermore, C_i clearly implies A_{i+1} for $i = 1, ..., n-1$. So, since we assumed that A_1 holds, we can distinguish the following two cases:

I. either $C_1, ..., C_n$ hold;

II. or there is a $k \in \{1, ..., n\}$ such that $C_1, ..., C_{k-1}$ and B_k hold.

We consider these two cases in turn.

I. Suppose that $C_1, ..., C_n$ hold, so that:

$$\begin{array}{rcll} T_i & \underline{\leftrightarrow} & R_i(T_{i+1} + U_{i+1}) & i = 1, ..., n-1 \\ T_n & \underline{\leftrightarrow} & R_n(T_{n+1} + U_1) & \\ U_i X & \underline{\leftrightarrow} & S_i & i = 1, ..., n \\ T_{n+1} X & \underline{\leftrightarrow} & R_1 \mathrm{shift}(Y) \ . & \end{array}$$

By assumption $T_1 X \underline{\leftrightarrow} R_1 \mathrm{shift}(Y)$, so the last equivalence implies $T_{n+1} X \underline{\leftrightarrow} T_1 X$. Then Lem. 2.5 yields $T_{n+1} \underline{\leftrightarrow} T_1$. To complete the proof for this case, it is sufficient to show that the equivalences

$$\begin{array}{rcll} T_i & \underline{\leftrightarrow} & R_i(T_{i+1} + U_{i+1}) & i = 1, ..., n-1 \\ T_n & \underline{\leftrightarrow} & R_n(T_1 + U_1) & \end{array}$$

together yield the following equivalence:

$$T_1 \quad \underline{\leftrightarrow} \quad R_1(R_2, ..., R_n, R_1)^*(U_2, ..., U_n, U_1) \ .$$

The construction of the required bisimulation relation is not hard, and is left to the reader.

II. Suppose that there is a $k \in \{1, ..., n\}$ such that $C_1, ..., C_{k-1}$ and B_k hold. Then

$$\begin{array}{rcll} T_i & \underline{\leftrightarrow} & R_i(T_{i+1} + U_{i+1}) & i = 1, ..., k-1 \\ T_k & \underline{\leftrightarrow} & R_k & \\ T_iX & \underline{\leftrightarrow} & R_i\text{shift}^i(Y) & i = 2, ..., k \\ U_iX & \underline{\leftrightarrow} & S_i & i = 2, ..., k \end{array}$$

and we are done.

The proof of the lemma is now complete. □

3.4 The Completeness Theorem

We are finally in a position to prove the desired completeness result.

Theorem 3.12 (Completeness) *Let $X, Y \in \mathbb{T}(\text{BPA}^{me*}(A))$. If $X \underline{\leftrightarrow} Y$, then $\mathcal{E} \vdash X = Y$.*

Proof: First of all, note that, as each process term is provably equal to a basic term (Lem. 3.1), it is sufficient to show that bisimilar basic terms are provably equal. This we proceed to do by induction on the well-founded ordering $\sqsubset$ on $\mathbb{B}\times\mathbb{B}$. To this end, assume that X, Y are basic terms such that $X \underline{\leftrightarrow} Y$. Suppose furthermore, as inductive hypothesis that if two basic terms X', Y' with $(X', Y') \sqsubset (X, Y)$ are bisimilar, then $\mathcal{E} \vdash X' = Y'$. We proceed to show that $\mathcal{E} \vdash X = Y$. For notational convenience, we shall write $X = Y$ in lieu of $\mathcal{E} \vdash X = Y$. Throughout the proof, we shall use the fact that the collection of basic terms is closed under transitions (Lem. 3.2) without further mention. We prove the claim by considering the following two cases:

Case 1. $X \notin \mathbb{H}$ or $Y \notin \mathbb{H}$;

Case 2. Both X and Y are in $\mathbb{H}$.

We examine these two cases in turn.

- Case 1. Assume that $X \notin \mathbb{H}$ or $Y \notin \mathbb{H}$. By symmetry, we may suppose that $X \notin \mathbb{H}$. Since $X \underline{\leftrightarrow} Y$, by possibly using axiom A3 we can adapt the expansions of X and Y to the following forms:

$$X = \sum_{i=1}^{m} \alpha_i X_i + \sum_{j=1}^{n} \beta_j \qquad Y = \sum_{i=1}^{m} \alpha_i Y_i + \sum_{j=1}^{n} \beta_j$$

where $X_i \underline{\leftrightarrow} Y_i$ for $i = 1, ..., m$. As $X \notin \mathbb{H}$, Lem. 3.8 gives that $X_i \prec X$ for $i = 1, ..., m$. Furthermore, again using Lem. 3.8, we infer that, for $i = 1, ..., m$, either $Y_i \prec Y$ or $Y_i \cong Y$. Therefore, for every index i, $(X_i, Y_i) \sqsubset (X, Y)$, and we may apply the inductive hypothesis to the equivalence $X_i \underline{\leftrightarrow} Y_i$ to derive that $X_i = Y_i$. Hence, by substitutivity it follows that $X = Y$, and we are done.

- Case 2. Assume that $X, Y \in \mathbb{H}$. We proceed with the proof by considering the possible forms these bisimilar terms may take. By symmetry, it is sufficient to distinguish the following three cases:

CASE 2.1. $X \equiv (P_1, ..., P_m)^*(Q_1, ..., Q_m)$ and $Y \equiv (R_1, ..., R_n)^*(S_1, ..., S_n)$;

CASE 2.2. $X \equiv P'(P_1, ..., P_m)^*(Q_1, ..., Q_m)$ for some proper derivative P' of P_m, and $Y \equiv (R_1, ..., R_n)^*(S_1, ..., S_n)$;

CASE 2.3. $X \equiv P'(P_1, ..., P_m)^*(Q_1, ..., Q_m)$ for some proper derivative P' of P_m, and $Y \equiv R'(R_1, ..., R_n)^*(S_1, ..., S_n)$ for some proper derivative R' of R_n.

We proceed by considering each of these cases in turn.

* CASE 2.1. Let $X \equiv (P_1, ..., P_m)^*(Q_1, ..., Q_m)$ and $Y \equiv (R_1, ..., R_n)^*(S_1, ..., S_n)$. By symmetry, it is sufficient to consider the following two sub-cases:

CASE 2.1.1. There exist a proper derivative Q' of some Q_i with $Q' \underline{\leftrightarrow} Y$, and a proper derivative S' of some S_j with $S' \underline{\leftrightarrow} X$;

CASE 2.1.2. No proper derivative of any Q_i is bisimilar to Y.

We consider these two sub-cases in turn.

- CASE 2.1.1. Assume that there exist a proper derivative Q' of some Q_i with $Q' \underline{\leftrightarrow} Y$, and a proper derivative S' of some S_j with $S' \underline{\leftrightarrow} X$.

Note that, by transitivity, we may infer that $Q' \underline{\leftrightarrow} S'$. As $Q' \prec X$ and $S' \prec Y$ (Lem. 3.6), we may apply the inductive hypothesis to each of the aforementioned equivalences to derive that

$$X = S' = Q' = Y$$

and we are done.

- CASE 2.1.2. Assume that no proper derivative of any Q_i is bisimilar to Y.

We begin the proof for this case by adapting X and Y.

Definition 3.13 *For $Z \equiv (T_1, ..., T_j)^*(U_1, ..., U_j)$, let $K(Z)$ denote the number of i's in $\{1, ..., j\}$ for which $\mathrm{shift}^i(Z) \underline{\leftrightarrow} Z$. Note that $K(Z) \geq 1$, as $\mathrm{shift}^j(Z) \equiv Z$.*

Put

$$\begin{array}{rcl} X_0 & \equiv & ((P_1, ..., P_m)^{K(Y)})^*((Q_1, ..., Q_m)^{K(Y)}) \\ Y_0 & \equiv & ((R_1, ..., R_n)^{K(X)})^*((S_1, ..., S_n)^{K(X)}) \ . \end{array}$$

It is not hard to see that, for every $i \in \{1, \ldots, m\}$ and $k \in \{0, \ldots, K(Y) - 1\}$,

$$\mathrm{shift}^i(X) \underline{\leftrightarrow} \mathrm{shift}^{i+km}(X_0) \ .$$

Similarly, for every $j \in \{1, \ldots, n\}$ and $k \in \{0, \ldots, K(X) - 1\}$,

$$\mathrm{shift}^j(Y) \underline{\leftrightarrow} \mathrm{shift}^{j+kn}(Y_0) \ .$$

Since $X_0 \underline{\leftrightarrow} X$ and $Y_0 \underline{\leftrightarrow} Y$, it follows that both $K(X_0)$ and $K(Y_0)$ are equal to $K(X) \cdot K(Y)$. Owing to axiom MEI5, the equalities $X = X_0$ and $Y = Y_0$ are provable. The rest of the proof for this case will be devoted to proving $X_0 = Y_0$. For notational convenience, put $X_0 \equiv (P_1, ..., P_M)^*(Q_1, ..., Q_M)$ and $Y_0 \equiv (R_1, ..., R_N)^*(S_1, ..., S_N)$.

We shall now prove that there exists an increasing sequence of integers $0 = c_0 < \cdots < c_M = N$ such that for $l = 0, ..., M$:

A_l $\mathrm{shift}^l(X_0) \underline{\leftrightarrow} \mathrm{shift}^{c_l}(Y_0)$;

B_l $\mathrm{shift}^i(Y_0) \not\underline{\leftrightarrow} Y_0$ for $c_{l-1} < i < c_l$ (where, by convention, $c_{-1} \triangleq 0$);

C_l $X_0 = (R_1, ..., R_{c_l}, P_{l+1}, ..., P_M)^*(S_1, ..., S_{c_l}, Q_{l+1}, ..., Q_M)$.

In fact, we shall show that the conjunction of the statements A_l, B_l and C_l implies the conjunction of A_{l+1}, B_{l+1} and C_{l+1} for $l < M$. Since A_0 (viz. $X_0 \leftrightarrow Y_0$), B_0 (viz. a vacuous statement) and C_0 (viz. $X_0 = X_0$) hold, we can then conclude that C_M (viz. $X_0 = Y_0$) holds, which is what we want to prove. Throughout the proof, we shall make use of the fact that, as $X \cong X_0$ and $Y \cong Y_0$, these terms dominate the same basic terms with respect to the well-founded ordering $\prec$. This remark will allow us to apply the inductive hypothesis to all pairs (X', Y') of bisimilar terms such that $(X', Y') \sqsubset (X_0, Y_0)$, and will be used without further mention.

Assume now that, for some l with $0 \leq l < M$, the statements A_i, B_i and C_i hold for all $i \leq l$ and some increasing sequence of integers $0 = c_0 < \cdots < c_l < N$. We set out to deduce the statements A_{l+1}, B_{l+1} and C_{l+1} for some c_{l+1} with $c_l < c_{l+1} \leq N$, where c_{l+1} equals N if and only if $l + 1 = M$.

First, we spell out the expansions of P_{l+1} and R_{c_l+1}:

$$P_{l+1} = \sum_{i \in I} T_i \qquad R_{c_l+1} = \sum_{j \in J} U_j$$

where the summands T_i and U_j are of the form either αV or α.

As A_l holds, $P_{l+1}\mathrm{shift}^{l+1}(X_0) + Q_{l+1} \leftrightarrow R_{c_l+1}\mathrm{shift}^{c_l+1}(Y_0) + S_{c_l+1}$. Hence, for every $i \in I$, either there exists an index $j \in J$ such that

$$T_i\mathrm{shift}^{l+1}(X_0) \quad \leftrightarrow \quad U_j\mathrm{shift}^{c_l+1}(Y_0) \tag{2}$$

or there exists a summand $\alpha S'$ of S_{c_l+1} such that

$$T_i\mathrm{shift}^{l+1}(X_0) \quad \leftrightarrow \quad \alpha S' \ . \tag{3}$$

Thus, I can be divided into the following, not necessarily disjoint, subsets:

$$\begin{array}{lcl} I_0 & \triangleq & \{i \in I \mid \exists \alpha, S' : \ S_{c_l+1} \xrightarrow{\alpha} S' \text{ and (3) holds}\} \\ I_1 & \triangleq & \{i \in I \mid \exists j \in J \text{ such that (2) holds}\} \ . \end{array}$$

On the other hand, for every $j \in J$ there exists an $i \in I$ such that equivalence (2) holds. In fact, suppose, towards a contradiction, that for some $j \in J$ this does not hold. Since $P_{l+1}\mathrm{shift}^{l+1}(X_0) + Q_{l+1} \leftrightarrow R_{c_l+1}\mathrm{shift}^{c_l+1}(Y_0) + S_{c_l+1}$, it follows that for this j there exists a summand $\alpha Q'$ of Q_{l+1} such that

$$\alpha Q' \quad \leftrightarrow \quad U_j\mathrm{shift}^{c_l+1}(Y_0) \ .$$

Since Y_0 is a derivative of $\mathrm{shift}^{c_l+1}(Y_0)$, then there is a proper derivative of Q_{l+1} that is bisimilar to Y_0, and therefore to Y. This contradicts the assumption for case 2.1.2.

Note that, by our previous reasoning, the index set I_1 is non-empty; let V_1 denote the term $\sum_{i \in I_1} T_i$. The following equation follows by possibly using axiom A3:

$$P_{l+1} \quad = \quad \sum_{i \in I_0} T_i + V_1 \ . \tag{4}$$

Moreover, the following equivalence is an immediate consequence of the fact that for every $j \in J$ there exists an $i \in I$ such that equivalence (2) holds:

$$V_1\mathrm{shift}^{l+1}(X_0) \quad \leftrightarrow \quad R_{c_l+1}\mathrm{shift}^{c_l+1}(Y_0) \ . \tag{5}$$

We proceed with the proof by deriving the following equality:

$$\sum_{i \in I_0} T_i \mathrm{shift}^{l+1}(X_0) + Q_{l+1} \quad = \quad S_{c_l+1} \ . \tag{6}$$

Proof of Equation (6). We show that each summand of S_{c_l+1} is provably equal to a summand of the term at the left-hand side of the equation, and vice versa.

Let $\alpha S'$ be a summand of S_{c_l+1}; then the transition $\mathrm{shift}^{c_l}(Y_0) \xrightarrow{\alpha} S'$ holds. As

$$P_{l+1}\mathrm{shift}^{l+1}(X_0) + Q_{l+1} \leftrightarrow \mathrm{shift}^{c_l}(Y_0)$$

by assumption A_l, either there exists an $i \in I_0$ such that (3) holds, or there exists a summand $\alpha Q'$ of Q_{l+1} such that

$$Q' \quad \leftrightarrow \quad S' \ . \tag{7}$$

If there exists an $i \in I_0$ such that (3) holds, then

1. either $T_i \equiv \alpha P'$ for some proper derivative P' of P_{l+1} such that $P'\mathrm{shift}^{l+1}(X_0) \leftrightarrow S'$,
2. or $T_i \equiv \alpha$ and $\mathrm{shift}^{l+1}(X_0) \leftrightarrow S'$.

As $S' \prec Y_0$ (Lem. 3.6), $P'\mathrm{shift}^{l+1}(X_0) \cong X_0$ and $\mathrm{shift}^{l+1}(X_0) \cong X_0$, in each of these two cases we may apply the inductive hypothesis and substitutivity to infer that

$$T_i\mathrm{shift}^{l+1}(X_0) \quad = \quad \alpha S' \ .$$

If (7) holds, then $Q' \prec X_0$ and $S' \prec Y_0$ (Lem. 3.6), so that we may apply the inductive hypothesis and substitutivity to infer

$$\alpha Q' \quad = \quad \alpha S' \ .$$

If α is a summand of S_{c_l+1}, then α must also be a summand of Q_{l+1}, and therefore of the term on the left-hand side of the equation. Hence, every summand of S_{c_l+1} is provably equal to a summand of the term at the left-hand side of (6).

By the symmetric argument it can be shown that every summand of the term at the left-hand side of (6) is provably equal to a summand of S_{c_l+1}. Namely, by definition of I_0 each term $T_i\mathrm{shift}^{l+1}(X_0)$ for $i \in I_0$ is bisimilar to a summand of S_{c_l+1}. Moreover, as $P_{l+1}\mathrm{shift}^{l+1}(X_0) + Q_{l+1} \leftrightarrow R_{c_l+1}\mathrm{shift}^{c_l+1}(Y_0) + S_{c_l+1}$, Y_0 is a derivative of $\mathrm{shift}(Y_0)$, and no derivative of Q_{l+1} is bisimilar to Y_0, every summand of Q_{l+1} must be bisimilar to a summand of S_{c_l+1}.

End of Proof of Equation (6).

We now proceed with our argument by analyzing equivalence (5) using Lem. 3.11. According to this lemma two possibilities may arise, which we consider in turn.

- Case 2.1.2.1. If the first case of Lem. 3.11 holds with respect to equivalence (5), then, in particular, there exists a term W such that

$$W\mathrm{shift}^{l+1}(X_0) \quad \leftrightarrow \quad S_{c_l+1} \ .$$

Equality (6) implies the equivalence

$$W\mathrm{shift}^{l+1}(X_0) \leftrightarrow \sum_{i \in I_0} T_i\mathrm{shift}(X_0) + Q_{l+1} \ .$$

Since X_0 is a derivative of $\mathrm{shift}^{l+1}(X_0)$ and $Y_0 \leftrightarrow X_0$, it follows that there is a proper derivative of Q_{l+1} which is bisimilar to Y_0, and therefore to Y. This contradicts the assumption at the start of case 2.1.2.

- Case 2.1.2.2. In light of the previous discussion, we may assume that the second case of Lem. 3.11 holds with respect to equivalence (5). In this case, note, first of all, that, for every $j, h \geq 0$,
$$\text{shift}^j(Y_0) \equiv \text{shift}^{j+hn}(Y_0) \ .$$
Therefore, by the second case of Lem. 3.11, there is an integer $k \in \{1, ..., n\}$ and there exist terms $V_2, ..., V_k$ and $W_2, ..., W_k$, whose proper derivatives are contained in those of V_1, such that:

$$\begin{array}{llcll}
(8) & V_i & \leftrightarrow & R_{c_l+i}(V_{i+1}+W_{i+1}) & i = 1, ..., k-1 \\
(9) & V_k & \leftrightarrow & R_{c_l+k} & \\
(10) & V_i\text{shift}^{l+1}(X_0) & \leftrightarrow & R_{c_l+i}\text{shift}^{c_l+i}(Y_0) & i = 2, ..., k \\
(11) & W_i\text{shift}^{l+1}(X_0) & \leftrightarrow & S_{c_l+i} & i = 2, ..., k \ .
\end{array}$$

Put $c_{l+1} \triangleq c_l + k$. We now show that A_{l+1}, B_{l+1} and C_{l+1} hold, and that $c_{l+1} \leq N$, with $c_{l+1} = N$ iff $l+1 = M$.

Equivalence (9) and the instance of (10) for $i = k$, together with Lem. 2.5, yield the equivalence
$$\text{shift}^{l+1}(X_0) \leftrightarrow \text{shift}^{c_{l+1}}(Y_0)$$
which corresponds to A_{l+1}.

We now show that B_{l+1} holds. Suppose, towards a contradiction, that $\text{shift}^{c_l+i}(Y_0) \leftrightarrow Y_0$ for some $i \in \{1, ..., k-1\}$. Equivalences (10) and (11), together with A4 and MEI1, imply that $\text{shift}^{c_l+i}(Y_0) \leftrightarrow (V_{i+1} + W_{i+1})\text{shift}^{l+1}(X_0)$. Therefore, as $X_0 \leftrightarrow Y_0$,
$$P_1\text{shift}(X_0) + Q_1 \leftrightarrow Y_0 \leftrightarrow \text{shift}^{c_l+i}(Y_0) \leftrightarrow (V_{i+1} + W_{i+1})\text{shift}^{l+1}(X_0) \ .$$
As X_0 is a derivative of $\text{shift}^{l+1}(X_0)$, and $Y_0 \leftrightarrow X_0$, it follows that Q_1 has a derivative bisimilar to Y_0, and therefore to Y. This contradicts the assumption for case 2.1.2. So it must be the case that $\text{shift}^{c_l+i}(Y_0) \not\leftrightarrow Y_0$ for $i = 1, ..., k-1$, which corresponds to B_{l+1}.

As $c_l < N$ and B_{l+1} holds, it follows that $c_l + i \neq N$ for $i = 1, ..., k-1$. Hence $c_{l+1} \leq N$. We now show that $c_{l+1} = N$ iff $l+1 = M$. By definition of $K(Z)$ (Def. 3.13) we have:

  - $c_{l+1} = N$ iff there are $K(Y_0)$ indices $i \in \{1, ..., c_{l+1}\}$ with $\text{shift}^i(Y_0) \leftrightarrow Y_0$;
  - $l+1 = M$ iff there are $K(X_0)$ indices $j \in \{1, ..., l+1\}$ with $\text{shift}^j(X_0) \leftrightarrow X_0$.

Recall that X_0 and Y_0 where designed such that $K(X_0) = K(Y_0)$. Furthermore, A_i and B_i for $i \leq l+1$ imply that $\text{shift}^i(Y_0) \leftrightarrow Y_0$ for $1 \leq i \leq c_{l+1}$ iff $i = c_j$ for some $j \in \{1, ..., l+1\}$ with $\text{shift}^j(X_0) \leftrightarrow X_0$. Hence, it follows that $c_{l+1} = N$ iff $l+1 = M$.

It remains to prove that C_{l+1} holds. Equality (4) implies that $L(V_1) \leq L(P_{l+1})$, so, by Lem. 2.7, $L(V_1) < L(X_0)$. Since the proper derivatives of the terms V_i are contained in those of V_1, it follows that $L(V_i) < L(X_0)$ for $i = 2, ..., k$. Now, invariance of L-value under bisimulation (Lem. 2.7), together with the equivalences $X_0 \leftrightarrow Y_0$, (8) and (9), yields

  1. $L(R_{c_l+i}(V_{i+1} + W_{i+1})) < L(Y_0)$ for $i = 1, ..., k-1$ and
  2. $L(R_{c_{l+1}}) < L(Y_0)$.

It follows that

  - $V_i \prec X_0$ for $i = 1, ..., k$,
  - $R_i(V_{i+1} + W_{i+1}) \prec Y_0$ for $i = 1, ..., k-1$ and
  - $R_{c_{l+1}} \prec Y_0$.

Thus we can apply induction to equivalences (8) and (9) to derive

$$(12) \qquad V_i \;=\; R_{c_l+i}(V_{i+1}+W_{i+1}) \qquad i=1,...,k-1$$
$$(13) \qquad V_k \;=\; R_{c_{l+1}} \;.$$

Furthermore, since $S' \prec Y_0$ for every proper derivative S' of each term S_{c_l+i} (Lem. 3.6), and since each derivative in the expansion of $W_i\mathrm{shift}^{l+1}(X_0)$ is $\cong$-equivalent to X_0, we can apply the same reasoning used in the proof of (6) to equivalence (11) to obtain

$$(14) \qquad W_i\mathrm{shift}^{l+1}(X_0) \;=\; S_{c_l+i} \qquad i=2,...,k \;.$$

The equations that we derived up to now can be used to prove

$$(15)\;\; X_0 \;=\; (R_1,...,R_{c_l+i-1},V_i,P_{l+2},...,P_M)^*(S_1,...,S_{c_l+i-1},S_{c_l+i},Q_{l+2},...,Q_M)$$

for $i=1,...,k-1$, and

$$(16) \qquad X_0 \;=\; (R_1,...,R_{c_{l+1}},P_{l+2},...,P_M)^*(S_1,...,S_{c_{l+1}},Q_{l+2},...,Q_M) \;.$$

In order to derive these equations, we apply induction on i. First, we deal with the case $i=1$:

$$\begin{aligned} X_0 &\overset{C_l}{=} (R_1,...,R_{c_l},P_{l+1},...,P_M)^*(S_1,...,S_{c_l},Q_{l+1},...,Q_M) \\ &\overset{(4)}{=} (R_1,...,R_{c_l},\sum_{i\in I_0}T_i+V_1,P_{l+2},...,P_M)^*(S_1,...,S_{c_l},Q_{l+1},Q_{l+2}...,Q_M) \\ &\overset{\mathrm{MEI3}'}{=} (R_1,...,R_{c_l},V_1,P_{l+2},...,P_M)^*(S_1,...,S_{c_l},\sum_{i\in I_0}T_iZ+Q_{l+1},Q_{l+2}...,Q_M) \end{aligned}$$

where

$$Z \overset{(4)}{=} (P_{l+2},...,P_M,R_1,...,R_{c_l},P_1)^*(Q_{l+2}...,Q_M,S_1,...,S_{c_l},Q_{l+1}) \;.$$

Applying MEI1 to the right-hand side of the above equation for $M-l-1$ times, followed by equation C_l, we obtain

$$Z \;=\; P_{l+2}(\cdots(P_MX_0+Q_M)\cdots)+Q_{l+2} \;.$$

Again applying MEI1 to the above equation, but this time from right to left, we deduce the equality $Z=\mathrm{shift}^{l+1}(X_0)$. Therefore:

$$\begin{aligned} X_0 &= \\ &(R_1,...,R_{c_l},V_1,P_{l+2},...,P_M)^*(S_1,...,S_{c_l},\sum_{i\in I_0}T_i\mathrm{shift}^{l+1}(X_0)+Q_{l+1},Q_{l+2}...,Q_M) \\ &\overset{(6)}{=} (R_1,...,R_{c_l},V_1,P_{l+2},...,P_M)^*(S_1,...,S_{c_l},S_{c_l+1},Q_{l+2}...,Q_M) \;. \end{aligned}$$

If $k>1$, then we have proven equation (15) for $i=1$. If $k=1$, then $c_{l+1}=c_l+1$, and by equation (13) $V_1=R_{c_{l+1}}$, which proves (16).

Next, suppose that we have proven equation (15) for $i=j$, where $1\leq j<k$. We derive either equation (15) for $i=j+1$, if $k>j+1$, or equation (16), if $k=j+1$. To this end, we reason as follows:

$$\begin{aligned} X_0 &\overset{(15),(12)}{=} \\ &(R_1,...,R_{c_l+j-1},R_{c_l+j}(V_{j+1}+W_{j+1}),P_{l+2},...,P_M)^*(S_1,...,S_{c_l+j},Q_{l+2},...,Q_M) \\ &\overset{\mathrm{MEI4}'}{=} (R_1,...,R_{c_l+j},V_{j+1},P_{l+2},...,P_M)^*(S_1,...,S_{c_l+j},W_{j+1}Z,Q_{l+2},...,Q_M) \end{aligned}$$

where

$$Z \stackrel{(12)}{=} (P_{l+2}, ..., P_M, R_1, ..., R_{c_l+j-1}, V_j)^*(Q_{l+2} ..., Q_M, S_1, ..., S_{c_l+j-1}, S_{c_l+j}) \ .$$

Applying MEI1 for $M-l-1$ times, followed by equation (15), we obtain

$$Z \ = \ P_{l+2}(\cdots(P_M X_0 + Q_M)\cdots) + Q_{l+2} \ .$$

Again using MEI1, we may now deduce the equality $Z = \mathrm{shift}^{l+1}(X_0)$. Therefore:

$$\begin{aligned} X_0 = & \\ & (R_1, ..., R_{c_l+j}, V_{j+1}, P_{l+2}, ..., P_M)^*(S_1, ..., S_{c_l+j}, W_{j+1}\mathrm{shift}^{l+1}(X_0), Q_{l+2}, ..., Q_M) \\ \stackrel{(14)}{=} & (R_1, ..., R_{c_l+j}, V_{j+1}, P_{l+2}, ..., P_M)^*(S_1, ..., S_{c_l+j}, S_{c_l+j+1}, Q_{l+2}, ..., Q_M) \ . \end{aligned}$$

If $k > j+1$, then we have proven equation (15) for $i = j+1$. If $k = j+1$, then $c_{l+1} = c_l + j + 1$, and by equation (13) $V_{j+1} = R_{c_{l+1}}$, which proves equation (16).

Equation (16) corresponds to C_{l+1}. Thus, we have completed the proof that the conjunction of A_l, B_l and C_l implies the conjunction of A_{l+1}, B_{l+1} and C_{l+1} for $0 \leq l < M$. Since A_0, B_0 and C_0 hold, we can then conclude C_M, that is, $X_0 = Y_0$. By axiom MEI5, it follows that $X = X_0$ and $Y = Y_0$. Therefore $X = Y$, which finishes the proof of this case.

∗ Case 2.2. Assume that $X \equiv P'(P_1, ..., P_m)^*(Q_1, ..., Q_m)$ for some proper derivative P' of P_m, and $Y \equiv (R_1, ..., R_n)^*(S_1, ..., S_n)$. For notational convenience we put $X' \equiv (P_1, ..., P_m)^*(Q_1, ..., Q_m)$. We proceed with the proof by distinguishing two sub-cases, depending on whether any of the terms Q_i has a proper derivative that is bisimilar to Y or not.

∘ Case 2.2.1. Suppose that there is a proper derivative Q' of some Q_i, for $i = 1, ..., m$, such that $Q' \leftrightarrow Y$.

As $P'X' \leftrightarrow R_1\mathrm{shift}(Y) + S_1$, and X is a derivative of X', it follows that S_1 has a proper derivative S' that is bisimilar to X. Hence, transitivity yields $X \leftrightarrow S' \leftrightarrow Q' \leftrightarrow Y$. Since $S' \prec Y$ and $Q' \prec X$ (Lem. 3.6), we can apply induction to these three equivalences to obtain

$$X = S' = Q' = Y \ .$$

∘ Case 2.2.2. Suppose that no proper derivative of any Q_i is bisimilar to Y.

As $P'X' \leftrightarrow Y \leftrightarrow R_1\mathrm{shift}(Y) + S_1$, there exist terms T_1 and U_1, whose proper derivatives are contained in those of P', such that:

$$P' \leftrightarrow T_1 + U_1 \tag{17}$$

$$T_1X' \leftrightarrow R_1\mathrm{shift}(Y) \tag{18}$$

$$U_1X' \leftrightarrow S_1 \ . \tag{19}$$

As $L(P') < L(X)$ (Lem. 2.7), in light of the equivalences $X \leftrightarrow Y$ and (17), invariance of L-value under bisimulation (Lem. 2.7) yields $L(T_1 + U_1) < L(Y)$. It follows that $P' \prec X'$ and $T_1 + U_1 \prec Y$. Therefore, by induction, equivalence (17) can be proven:

$$P' \ = \ T_1 + U_1 \ . \tag{20}$$

Furthermore, since $S' \prec Y$ for every proper derivative S' of S_1 (Lem. 3.6), and since each derivative in the expansion of U_1X' is $\cong$-equivalent to X, we can apply the same reasoning used in the proof of (6) to equivalence (19) to obtain

$$U_1X' \ = \ S_1 \ . \tag{21}$$

Finally, we can apply Lem. 3.11 to equivalence (18) to obtain the following two possibilities:

- Case 2.2.2.1. If case I of Lem. 3.11 holds, then there are terms $V_1, ..., V_n$, whose proper derivatives are contained in those of T_1, such that:

$$
\begin{array}{llll}
(22) & T_1 & \underline{\leftrightarrow} & R_1(R_2, ..., R_n, R_1)^*(V_2, ..., V_n, V_1) \\
(23) & V_i X' & \underline{\leftrightarrow} & S_i \qquad\qquad i = 1, ..., n \ .
\end{array}
$$

By (17) and Lem. 2.7, it follows that $L(T_1) \leq L(P') < L(X)$. In light of the equivalences $X \underline{\leftrightarrow} Y$ and (22), invariance of L-value under bisimulation (Lem. 2.7), yields that

$$L(R_1(R_2, ..., R_n, R_1)^*(V_2, ..., V_n, V_1)) < L(Y) \ .$$

Hence, we can apply induction to equivalence (22) to obtain

$$
(24) \qquad T_1 \ = \ R_1(R_2, ..., R_n, R_1)^*(V_2, ..., V_n, V_1) \ .
$$

Moreover, since $S' \prec Y$ for every proper derivative S' of S_i (Lem. 3.6), and since each derivative in the expansion of $V_i X'$ is $\cong$-equivalent to X, we can apply the same reasoning used in the proof of (6) to derive equivalence (23):

$$
(25) \qquad V_i X' \ = \ S_i \qquad\qquad i = 1, ..., n \ .
$$

Hence, we may conclude the proof for this case thus:

$$
\begin{array}{rcl}
X \ \equiv \ P'X' & \overset{(20)}{=} & (T_1 + U_1)X' \\
& \overset{\text{A4},(24),(21)}{=} & R_1(R_2, ..., R_n, R_1)^*(V_2, ..., V_n, V_1)X' + S_1 \\
& \overset{\text{MEI2},(25)}{=} & R_1(R_2, ..., R_n, R_1)^*(S_2, ..., S_n, S_1) + S_1 \\
& \overset{\text{MEI1}}{=} & Y \ .
\end{array}
$$

- Case 2.2.2.2. If case II of Lem. 3.11 holds, then there is a $k \in \{1, ..., n\}$ and there are terms $T_2, ..., T_k$ and $U_2, ..., U_k$, whose proper derivatives are contained in those of T_1, such that:

$$
\begin{array}{llll}
(26) & T_i & \underline{\leftrightarrow} & R_i(T_{i+1} + U_{i+1}) \qquad\qquad i = 1, ..., k-1 \\
(27) & T_k & \underline{\leftrightarrow} & R_k \\
(28) & T_k X' & \underline{\leftrightarrow} & R_k \text{shift}^k(Y) \\
(29) & U_i X' & \underline{\leftrightarrow} & S_i \qquad\qquad i = 2, ..., k \ .
\end{array}
$$

By (17) and Lem. 2.7, it follows that $L(T_1) \leq L(P') < L(X)$. Since the proper derivatives of the terms T_i are contained in those of T_1, it follows that $L(T_i) < L(X)$ for $i = 2, ..., k$. In light of the equivalences $X \underline{\leftrightarrow} Y$ and (26)–(27), invariance of L-value under bisimulation (Lem. 2.7) yields

- $L(R_i(T_{i+1} + U_{i+1})) < L(Y)$ for $i = 1, ..., k-1$ and
- $L(R_k) < L(Y)$.

Hence, we may apply induction to equivalences (26) and (27) to infer that:

$$
\begin{array}{llll}
(30) & T_i & = & R_i(T_{i+1} + U_{i+1}) \qquad\qquad i = 1, ..., k-1 \\
(31) & T_k & = & R_k \ .
\end{array}
$$

Reasoning as in the proof of (6), it is not hard to deduce equivalence (29):

$$(32) \qquad U_i X' \;=\; S_i \qquad\qquad i = 2, ..., k \ .$$

Furthermore, equivalences (27), (28), and Lem. 2.5 imply that $X' \underline{\leftrightarrow} \mathrm{shift}^k(Y)$. This equivalence can be deduced by case 2.1 of the proof:

$$(33) \qquad X' \;=\; \mathrm{shift}^k(Y) \ .$$

We can now use these equalities to derive $X = Y$ as follows:

$$\begin{array}{rcl}
X \;\equiv\; P'X' & \stackrel{(20)}{=} & (T_1 + U_1)X' \\
& \stackrel{\mathrm{A4},(30),(21)}{=} & R_1(T_2 + U_2)X' + S_1 \\
& \stackrel{(30),(32)}{=} & R_1(R_2(T_3 + U_3)X' + S_2) + S_1 \\
& \stackrel{(30),(32)}{=} & \cdots \\
& \stackrel{(31),(32)}{=} & R_1(R_2(\cdots(R_{k-1}(R_k X' + S_k) + S_{k-1})\cdots) + S_2) + S_1 \\
& \stackrel{(33)}{=} & R_1(R_2(\cdots(R_{k-1}(R_k \mathrm{shift}^k(Y) + S_k) + S_{k-1})\cdots) + S_2) + S_1 \ .
\end{array}$$

Finally, since $R_i \mathrm{shift}^i(Y) + S_i \stackrel{\mathrm{MEI1}}{=} \mathrm{shift}^{i-1}(Y)$ for $i = 1, ..., k$, we can apply axiom MEI1 k times in order to reduce this last term to Y. This completes the proof for case 2.2.

$*$ Case 2.3. Assume that $X \equiv P'(P_1, ..., P_m)^*(Q_1, ..., Q_m)$ for some proper derivative P' of P_m, and that $Y \equiv R'(R_1, ..., R_n)^*(S_1, ..., S_n)$, for some proper derivative R' of R_n. For convenience, put $X' \equiv (P_1, ..., P_m)^*(Q_1, ..., Q_m)$ and $Y' \equiv (R_1, ..., R_n)^*(S_1, ..., S_n)$.

We proceed with the proof by analyzing the equivalence $P'X' \underline{\leftrightarrow} R'Y'$ using Lem. 2.5. By symmetry, we may restrict ourselves to considering the following two sub-cases, depending on whether $|X'| = |Y'|$ or $|X'| < |Y'|$, respectively.

$\circ$ Case 2.3.1. Assume that $|X'| = |Y'|$. Then an application of Lem. 2.5 yields the equivalences $P' \underline{\leftrightarrow} R'$ and $X' \underline{\leftrightarrow} Y'$. As $L(P') < L(X)$ and $L(R') < L(Y)$ (Lem. 2.7), the inductive hypothesis yields $P' = R'$. To conclude the proof for this case, it is therefore sufficient to note that the equality $X' = Y'$ follows from $X' \underline{\leftrightarrow} Y'$ by case 2.1 of the proof.

$\circ$ Case 2.3.2. Assume that $|X'| < |Y'|$. Then an application of Lem. 2.5 yields a proper derivative P'' of P' such that

$$\begin{array}{rcl}
P' & \underline{\leftrightarrow} & R'P'' \\
P''X' & \underline{\leftrightarrow} & Y' \ .
\end{array}$$

As $L(P') < L(X)$ (Lem. 2.7), invariance of L-value under bisimulation (Lem. 2.7) yields $L(R'P'') < L(Y)$. Then we may apply the inductive hypothesis to deduce the equality $P' = R'P''$. Furthermore, $P''X' = Y'$ follows from the equivalence $P''X' \underline{\leftrightarrow} Y'$ by case 2.2 of the proof. Hence, $P'X' = R'P''X' = R'Y'$.

This completes the proof for case 2.3. As we have examined all the possible forms $X, Y \in \mathbb{H}$ may take, the proof for case 2 is complete.

We have therefore shown the completeness theorem. □

In light of Lem. 2.4, Thm. 3.12 has the following corollary.

Corollary 3.14 *The axiom system $\mathcal{E}$ is an ω-complete axiomatization of the algebra* $\mathrm{T}(\mathrm{BPA}^{me*}(A))$ *modulo bisimulation equivalence.*

4 An Expressiveness Hierarchy for Multi-Exit Iteration

As shown in [8], the addition of multi-exit iteration to BPA yields a language that, modulo bisimulation equivalence, is strictly more expressive than that obtained by augmenting BPA with the standard binary Kleene star. For example, if the set of actions A contains at least two elements, then the process $(a, a)^*(a, b)$ cannot be expressed, modulo bisimulation equivalence, in ACP [7], and *a fortiori* in BPA, enriched with the binary Kleene star (cf. Lem. 3.2.3 in *op. cit.*).

Let us say that a term of the form $(P_1, \ldots, P_m)^*(Q_1, \ldots, Q_n)$ has n-exit iteration. By analogy with the aforementioned result from [8], we shall now argue that, in the presence of a countably infinite set of actions, the sequence of k-exit iteration operations induces a hierarchy of super-languages of BPA with a strictly increasing expressive power modulo bisimulation equivalence. To this end, we shall show that, for every positive integer k, there is a process over $k+1$ distinct actions that can be specified using $(k+1)$-exit iteration, but not using h-exit iteration with $h \leq k$.

Before embarking in the proof of this fact, we introduce some notions that will be useful in our argument.

Definition 4.1 *A non-empty sequence of transitions*

$$P_0 \stackrel{\alpha_1}{\to} P_1 \stackrel{\alpha_2}{\to} P_2 \cdots \stackrel{\alpha_{n+1}}{\to} P_{n+1}$$

is called a loop *from P iff $P_0 \equiv P_{n+1} \equiv P$. The* termination actions *of a term P are those actions a such that $P \stackrel{a}{\to} \checkmark$. Finally, for every $k \geq 1$, we write* BPA^{k*} *for the set of terms in the language $\mathbb{T}(\mathrm{BPA}^{me*}(A))$ that may use h-exit iteration with $h \leq k$.*

The following lemma provides the key to our expressiveness result.

Lemma 4.2 *Let $k \geq 1$ and let P be a term in the language* BPA^{k*}*. Then every loop from P traverses at most k terms with distinct, non-empty sets of termination actions.*

Proof: By structural induction on P. We proceed by a case analysis on the form P may take. The case $P \equiv \alpha$ is obviously vacuous because actions and variables have no loop from them.

- Case: $P \equiv Q + R$. By Lem. 3.5, if P' is a proper derivative of $P \equiv Q + R$, then $g(P') < g(P)$. Moreover, again by Lem. 3.5, the function g is non-increasing with respect to transitions. It follows that a term of the form $Q + R$ has no loop from it. This case is therefore vacuous.
- Case: $P \equiv QR$. Consider a loop

$$P_0 \stackrel{\alpha_1}{\to} P_1 \stackrel{\alpha_2}{\to} P_2 \cdots \stackrel{\alpha_{n+1}}{\to} P_{n+1}$$

from P. Two possibilities may now arise:

1. there exists a loop
$$Q_0 \stackrel{\alpha_1}{\to} Q_1 \stackrel{\alpha_2}{\to} Q_2 \cdots \stackrel{\alpha_{n+1}}{\to} Q_{n+1}$$
from Q such that $P_i \equiv Q_i R$ for every $i = 0, \ldots, n+1$, or

2. there exist $l \in \{1, \ldots, n\}$ and sequences of transitions

$$Q \equiv Q_0 \stackrel{\alpha_1}{\to} Q_1 \stackrel{\alpha_2}{\to} Q_2 \cdots Q_{l-1} \stackrel{\alpha_l}{\to} \checkmark \quad \text{and} \quad R \equiv P_l \stackrel{\alpha_{l+1}}{\to} P_{l+1} \cdots \stackrel{\alpha_{n+1}}{\to} P_{n+1}$$

such that $P_i \equiv Q_i R$ for $i = 0, \ldots, l-1$.

If the first possibility applies, then all the terms traversed by the loop from P have no termination actions. If the second applies, then the loop from P traverses exactly the same terms visited by the following loop from R:

$$R \equiv P_l \stackrel{\alpha_{l+1}}{\to} P_{l+1} \cdots \stackrel{\alpha_{n+1}}{\to} P_{n+1} \equiv P \stackrel{\alpha_1}{\to} Q_1 R \stackrel{\alpha_2}{\to} Q_2 R \cdots Q_{l-1} R \stackrel{\alpha_l}{\to} R$$

and the claim follows immediately by induction.

- CASE: $P \equiv (Q_1, \ldots, Q_m)^*(R_1, \ldots, R_h)$, where $h \leq k$. Note, first of all, that no proper derivative of any of the terms R_i can be traversed by a loop from P (Lem. 3.5). Thus the only terms with non-empty sets of termination actions in loops from P are those of the form $\mathrm{shift}^i(P)$ for some non-negative integer i. There are at most h terms with this form that have distinct sets of termination actions.

□

As an immediate corollary of the above lemma, we now obtain the following result.

Corollary 4.3 *Let k be a positive integer. If the set of actions A contains at least $k+1$ distinct actions $a_1, \ldots, a_{k+1}$, then the process*

$$a_1{}^*(a_1, a_2, \ldots, a_{k+1})$$

cannot be specified in the language BPA^{k*} *modulo bisimulation equivalence.*

Proof: Immediate by Lem. 4.2, because the term under consideration has a loop that traverses $k+1$ terms with distinct, non-empty sets of termination actions. □

Let now the set of actions $A = \{a_1, a_2, \ldots\}$ be countably infinite. By Cor. 4.3, it follows that, for every $k \geq 1$, the process

$$a_1{}^*(a_1, a_2, \ldots, a_{k+1})$$

cannot be specified in the language BPA^{k*} modulo bisimulation equivalence. Thus, for every $k \geq 1$, the language $\mathrm{BPA}^{(k+1)*}$ is strictly more expressive than BPA^{k*} modulo bisimulation equivalence. This establishes the promised expressiveness hierarchy for the collection of languages BPA^{k*} ($k \geq 1$).

References

[1] L. ACETO, W. J. FOKKINK, R. VAN GLABBEEK, AND A. INGÓLFSDÓTTIR, *Axiomatizing prefix iteration with silent steps*, Research Report RS–95–56, BRICS (Basic Research in Computer Science, Centre of the Danish National Research Foundation), Department of Mathematics and Computer Science, Aalborg University, November 1995. To appear in *Information and Computation*. Available by anonymous ftp at the address `ftp.brics.aau.dk` in the directory `pub/BRICS/RS/95/56`.

[2] L. Aceto and J. F. Groote, *A complete equational axiomatization for MPA with string iteration*, Research Report RS–95–28, BRICS (Basic Research in Computer Science, Centre of the Danish National Research Foundation), Department of Mathematics and Computer Science, Aalborg University, May 1995. Available by anonymous ftp at the address `ftp.brics.aau.dk` in the directory `pub/BRICS/RS/95/28`.

[3] L. Aceto and A. Ingólfsdóttir, *A complete equational axiomatization for prefix iteration with silent steps*, Research Report RS–95–5, BRICS (Basic Research in Computer Science, Centre of the Danish National Research Foundation), Department of Mathematics and Computer Science, Aalborg University, Jan. 1995. To appear in the *Proceedings of AMAST '96*. Available by anonymous ftp at the address `ftp.daimi.aau.dk` in the directory `pub/BRICS/95/5`.

[4] J. Baeten, J. Bergstra, and J. Klop, *Decidability of bisimulation equivalence for processes generating context-free languages*, J. Assoc. Comput. Mach., 40 (1993), pp. 653–682.

[5] J. Baeten and J. Klop, eds., *Proceedings CONCUR 90, Amsterdam*, vol. 458 of Lecture Notes in Computer Science, Springer-Verlag, 1990.

[6] J. Baeten and C. Verhoef, *A congruence theorem for structured operational semantics*, in Best [12], pp. 477–492.

[7] J. Baeten and W. Weijland, *Process Algebra*, Cambridge Tracts in Theoretical Computer Science 18, Cambridge University Press, 1990.

[8] J. Bergstra, I. Bethke, and A. Ponse, *Process algebra with iteration*, Report CS-R9314, Programming Research Group, University of Amsterdam, 1993. Available by anonymous ftp at the address `fwi.uva.nl` as `pub/programming-research/reports/1993/P9314.ps.Z`.

[9] ——, *Process algebra with iteration and nesting*, Computer Journal, 37 (1994), pp. 243–258.

[10] J. Bergstra and J. Klop, *Fixed point semantics in process algebras*, Report IW 206, Mathematisch Centrum, Amsterdam, 1982.

[11] ——, *Algebra of communicating processes with abstraction*, Theoretical Comput. Sci., 37 (1985), pp. 77–121.

[12] E. Best, ed., *Proceedings CONCUR 93, Hildesheim, Germany*, vol. 715 of Lecture Notes in Computer Science, Springer-Verlag, 1993.

[13] D. Caucal, *Graphes canoniques de graphes algébriques*, Theoretical Informatics and Applications, 24 (1990), pp. 339–352.

[14] J. H. Conway, *Regular algebra and finite machines*, Chapman and Hall, 1971.

[15] I. Copi, C. Elgot, and J. Wright, *Realization of events by logical nets*, J. Assoc. Comput. Mach., 5 (1958), pp. 181–196.

[16] W. J. Fokkink, *A complete equational axiomatization for prefix iteration*, Inf. Process. Lett., 52 (1994), pp. 333–337.

[17] ——, *On the completeness of the equations for the Kleene star in bisimulation*, Logic Group Preprint Series 141, Dept. of Philosophy, Utrecht University, Sept. 1995. To appear in the *Proceedings of AMAST '96*. Available by anonymous ftp from `phil.ruu.nl` as `logic/PREPRINTS/preprint141.ps`.

[18] ——, *A complete axiomatization for prefix iteration in branching bisimulation*, Fundamenta Informaticae, 26 (1996), pp. 103–113.

[19] W. J. Fokkink and R. van Glabbeek, *Ntyft/ntyxt rules reduce to ntree rules*, *Information and Computation*, 126 (1996), pp. 1–10.

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

[21] ——, *Prefix iteration in basic process algebra: applying termination techniques*, in Proceedings ACP 95, Eindhoven, A. Ponse, C. Verhoef, and B. v. Vlijmen, eds., vol. Report CS-95-14, Eindhoven University of Technology, 1995, pp. 139–156.

[22] R. v. Glabbeek, *The linear time – branching time spectrum*, in Baeten and Klop [5], pp. 278–297.

[23] ——, *A complete axiomatization for branching bisimulation congruence of finite-state behaviours*, in Mathematical Foundations of Computer Science 1993, Gdansk, Poland, A. Borzyszkowski and S. Sokołowski, eds., vol. 711 of Lecture Notes in Computer Science, Springer-Verlag, 1993, pp. 473–484. Available by anonymous ftp from `Boole.stanford.edu`.

[24] ——, *The linear time – branching time spectrum II: the semantics of sequential processes with silent moves*, in Best [12], pp. 66–81.

[25] J. F. Groote, *A new strategy for proving ω–completeness with applications in process algebra*, in Baeten and Klop [5], pp. 314–331.

[26] R. Keller, *Formal verification of parallel programs*, Comm. ACM, 19 (1976), pp. 371–384.

[27] S. Kleene, *Representation of events in nerve nets and finite automata*, in Automata Studies, C. Shannon and J. McCarthy, eds., Princeton University Press, 1956, pp. 3–41.

[28] D. Kozen, *A completeness theorem for Kleene algebras and the algebra of regular events*, Information and Computation, 110 (1994), pp. 366–390.

[29] D. Krob, *Complete systems of B-rational identities*, Theoretical Comput. Sci., 89 (1991), pp. 207–343.

[30] R. Milner, *A Calculus of Communicating Systems*, vol. 92 of Lecture Notes in Computer Science, Springer-Verlag, 1980.

[31] ——, *A complete inference system for a class of regular behaviours*, J. Comput. System Sci., 28 (1984), pp. 439–466.

[32] ——, *Communication and Concurrency*, Prentice-Hall International, Englewood Cliffs, 1989.

[33] ——, *A complete axiomatisation for observational congruence of finite-state behaviors*, Information and Computation, 81 (1989), pp. 227–247.

[34] D. Park, *Concurrency and automata on infinite sequences*, in 5^{th} GI Conference, Karlsruhe, Germany, P. Deussen, ed., vol. 104 of Lecture Notes in Computer Science, Springer-Verlag, 1981, pp. 167–183.

[35] D. Perrin, *Finite automata*, in Handbook of Theoretical Computer Science, J. van Leeuwen, ed., vol. B: Formal Models and Semantics, Elsevier Science Publishers B.V., 1990, ch. 1, pp. 1–57.

[36] G. Plotkin, *A structural approach to operational semantics*, Report DAIMI FN-19, Computer Science Department, Aarhus University, 1981.

[37] A. Salomaa, *Two complete axiom systems for the algebra of regular events*, J. Assoc. Comput. Mach., 13 (1966), pp. 158–169.

[38] ——, *Theory of Automata*, vol. 100 of International Series of Monographs in Pure and Applied Mathematics (I.N. Sneddon and M. Stark eds.), Pergamon Press, Oxford, 1969.

[39] P. Sewell, *Bisimulation is not finitely (first order) equationally axiomatisable*, in Proceedings 9^{th} Annual Symposium on Logic in Computer Science, Paris, France, IEEE Computer Society Press, 1994, pp. 62–70.

[40] D. Troeger, *Step bisimulation is pomset equivalence on a parallel language without explicit internal choice*, Mathematical Structures in Computer Science, 3 (1993), pp. 25–62.

Recent Publications in the BRICS Report Series

RS-96-22 **Luca Aceto and Wan J. Fokkink.** ***An Equational Axiomatization for Multi-Exit Iteration*****. June 1996. 30 pp.**

RS-96-21 **Dany Breslauer, Tao Jiang, and Zhigen Jiang.** ***Rotation of Periodic Strings and Short Superstrings*****. June 1996. 14 pp.**

RS-96-20 **Olivier Danvy and Julia L. Lawall.** ***Back to Direct Style II: First-Class Continuations*****. June 1996. 36 pp. A preliminary version of this paper appeared in the proceedings of the 1992 ACM Conference on Lisp and Functional Programming, William Clinger, editor, LISP Pointers, Vol. V, No. 1, pages 299–310, San Francisco, California, June 1992. ACM Press.**

RS-96-19 **John Hatcliff and Olivier Danvy.** ***Thunks and the λ-Calculus*****. June 1996. 22 pp. To appear in** ***Journal of Functional Programming*****.**

RS-96-18 **Thomas Troels Hildebrandt and Vladimiro Sassone.** ***Comparing Transition Systems with Independence and Asynchronous Transition Systems*****. June 1996. 14 pp. To appear in*****Concurrency Theory: 7th International Conference*****, CONCUR '96 Proceedings, LNCS, 1996.**

RS-96-17 **Olivier Danvy, Karoline Malmkjær, and Jens Palsberg.** ***Eta-Expansion Does The Trick (Revised Version)*****. May 1996. 30 pp. To appear in** ***ACM Transactions on Programming Languages and Systems (TOPLAS)*****.**

RS-96-16 **Lisbeth Fajstrup and Martin Raußen.** ***Detecting Deadlocks in Concurrent Systems*****. May 1996. 10 pp.**

RS-96-15 **Olivier Danvy.** ***Pragmatic Aspects of Type-Directed Partial Evaluation*****. May 1996. 27 pp.**

RS-96-14 **Olivier Danvy and Karoline Malmkjær.** ***On the Idempotence of the CPS Transformation*****. May 1996. 15 pp.**

RS-96-13 **Olivier Danvy and René Vestergaard.** ***Semantics-Based Compiling: A Case Study in Type-Directed Partial Evaluation*****. May 1996. 28 pp.**