Process Algebra with Autonomous Actions

Marc Voorhoeve and Twan Basten

Eindhoven University of Technology
email:{wsinmarc,tbasten}@win.tue.nl

Abstract

This paper introduces autonomous observable actions into process algebra. These actions can be observed but cannot be controlled by an environment. The proposed extension of ACP allows verifications without silent steps and fairness assumptions. The inclusion of inequalities makes it possible to verify that an implementation satisfies a given specification, with the specification indicating exactly where the implementation may reduce nondeterminism.

1 Introduction

Process algebra studies *processes* that are built from *actions* and operators. Two kinds of actions have been extensively studied: the observable ones and the unobservable (internal or silent) action τ. An observable action can *occur* within a process if the process is ready to perform the action and some controlling *environment* allows it to occur. In contrast, the silent action τ may occur without any consent. In ACP, the process algebra studied in this paper, the operators by which processes can be controlled from an environment are *encapsulation* and *communication*. Observable actions can be disabled by an environment by means of encapsulation; silent actions cannot be disabled. Similarly, observable actions are capable of communicating with actions from an environment, whereas τ is not.

This paper presents *autonomous* actions, which are observable, but not controllable by an environment, thus sharing with τ the property that they can neither be encapsulated nor participate in a communication. The difference between autonomous actions and τ is that autonomous actions can be observed, whereas τ cannot. The fact that τ is unobservable means that an environment can only detect an occurrence of τ if the process possesses less options for continuation after the occurrence than before. This is reflected by equations for processes, such as the branching equations $B1$ and $B2$ in ACP that allow to eliminate silent actions. No such elimination is possible for autonomous actions, as each occurrence can be observed.

A process that is capable of performing an initial autonomous action is called *unstable*. A process in which no initial autonomous action can occur is called *stable*, as an environment can stop its progress by encapsulating its initial actions. A choice involving an unstable process is *nondeterministic*, since an environment cannot prevent the autonomous action from occurring. We call this kind of nondeterminism *unstable nondeterminism*. Another kind of nondeterminism arises when several non-autonomous actions with the same visible effect can occur, leading to different subsequent processes. We call this *stable nondeterminism*.

Instead of only considering process equivalence, a partial order $\leq$ on processes is introduced. If processes p and q satisfy $p \leq q$, then either both processes are equal, or both p and q are unstable and the behaviour of p is a subset of the behaviour of q. Since there must exist an autonomous action that both p and q can perform, and since this action cannot be controlled, an observer, when confronted with p, cannot be sure that he is not dealing with q instead. So $\leq$ is based on reduction of *unstable* nondeterminism.

Partial order algebras that are based on reduction of nondeterminism are important for software engineering. Often, design decisions are made that reduce nondeterminism in a specification. The advantage of the approach in this paper is that in the specification one can distinguish unstable nondeterminism that may be reduced and stable nondeterminism that may not.

The main inspiration of the present paper is [BaBe94], where CSP-like choice operators [Hoa85] are introduced in ACP. The "partial bisimulation" preorder on processes from [BaBe94] that is also used in the present paper is related to the preorders in [Abr87] and [Wal90] that deal with the notion of *divergence*: a possibly infinite chain of autonomous actions (in their case silent actions τ).

The reason for introducing CSP-like choice operators in ACP in [BaBe94] is that CSP distinguishes between deterministic and nondeterministic choice. Autonomous actions offer an alternative to nondeterministic choice operators, with the advantage of considerably simpler equations and models. Also, in this paper axioms and proof rules are given for reasoning algebraically with infinite processes, whereas the reasoning in [BaBe94] (and CSP for that matter) is model-based.

The authors wish to thank Jos Baeten and Pedro D'Argenio for their suggestions and corrections. Jan Bergstra is thanked for sharing with us his insights regarding communication and autonomous behaviour.

2 The Basic Algebra

2.1 Signature and Axioms

Our basic algebra $\mathrm{BPAaa}^{\leq}$ is an extension of BPA_{δ}: basic process algebra with inaction (c.f. [BaWe90]). The signature and axioms are given in Table 1. First, in the header, the sort of atoms (A) is declared, which is a parameter of the theory. Each atom is an action. An overlining operator creates from an atom an autonomous action. The sort F of autonomous actions is thus equal to $\{a : \mathsf{A} \bullet \overline{a}\}$ [1]. The set $\mathsf{A} \cup \mathsf{F}$ of all actions is denoted AC. Note that AC is only an abbreviation and not a sort. The actions a and $\overline{a}$ have the same visible effect and only differ in the way they can be controlled by an environment. The sort P of processes is a supersort of both A and F. Sorts and their subsort relation are introduced in the first table entry.

The operator $\bar{\cdot}$ as well as the standard BPA operators $\cdot + \cdot$ and $\cdot \cdot \cdot$ and the constant δ are introduced in the second table entry, followed by the declaration of some variables and the axioms of the theory. As usual, the sequencing operator $(\cdot)$ has priority over choice $(+)$ and may be omitted. The axiom system consists of equations and inequalities, giving rise to a partial order algebra (c.f. [Hen88]), which means that in reasoning with inequalities, one is allowed to use reflexivity, transitivity, antisymmetry, closedness under contexts and substitutivity. Note that, by antisymmetry, for two terms x and y, $x = y$ iff $x \leq y$ and $y \leq x$.

The I axioms (for "inequality") state that unstable nondeterminism may be reduced in an implementation. The algebra BPAaa is obtained from $\mathrm{BPAaa}^{\leq}$ by removing these inequalities from the axioms in Table 1. Note that BPAaa with atoms A equals BPA_{δ} with atoms AC.

The $\mathrm{BPAaa}^{\leq}$ terms can be normalized. We inductively define the set $\mathcal{B}$ of basic terms as follows.

$$\{\delta\} \cup \mathsf{AC} \subseteq \mathcal{B}, \quad \forall t : \mathcal{B}; e : \mathsf{AC} \bullet e \cdot t \in \mathcal{B}, \quad \forall s, t : \mathcal{B} \bullet s + t \in \mathcal{B}.$$

Property 1 *For every closed term $p : \mathsf{P}$, there exists a $t : \mathcal{B}$ such that* BPAaa $\vdash p = t$ *(and thus* $\mathrm{BPAaa}^{\leq} \vdash p = t$*).*

[1] We use the Z-style notation $\{a : A \bullet f(a)\}$ instead of $\{f(a) \mid a \in A\}$ throughout the paper

BPAaa$^{\leq}$(A)

$\mathsf{F}, \mathsf{P}\ ;\ \mathsf{A} \subseteq \mathsf{P}\ ;\ \mathsf{F} \subseteq \mathsf{P}$

$\delta : \mathsf{P} \qquad \overline{\bullet} : \mathsf{A} \to \mathsf{F} \qquad \bullet + \bullet,\ \bullet \cdot \bullet : \mathsf{P} \times \mathsf{P} \to \mathsf{P}$

$a : \mathsf{A}; x, y, z : \mathsf{P}$

$x + y = y + x$	$A1$	$x + \delta = x$	$A6$
$(x + y) + z = x + (y + z)$	$A2$	$\delta \cdot x = \delta$	$A7$
$x + x = x$	$A3$		
$(x + y)z = x \cdot z + y \cdot z$	$A4$	$\overline{a} \leq \overline{a} + x$	$I1$
$(x \cdot y)z = x(y \cdot z)$	$A5$	$\overline{a} \cdot x \leq \overline{a} \cdot x + y$	$I2$

Table 1: Basic process algebra with autonomous actions

Proof Standard term rewriting, using $A3 \ldots A7$ as rewrite rules from left to right (c.f. [BaVe95]). □

Modulo axioms $A1$ and $A2$, and using $A6$, a basic term can be represented as $\sum_{i:I} a_i + \sum_{j:J} b_j \cdot t_j$, where I, J are finite index sets, the a_i and b_j actions and the t_j again basic terms. An empty sum is equal to δ. By $A3$, it may be assumed that all summands are different.

Example The theory BPAaa$^{\leq}$ may be used to demonstrate equivalence of the following processes that describe login procedures. The first process $L1$ prompts for a user identification and, having received one, prompts for a password. After having checked the combination, access is granted or refused. The user of the process perceives the prompting, granting and refusing as autonomous actions he cannot influence. The other actions are interactions between the process and the user. So $L1$ can be described as $\overline{pui} \cdot ui \cdot M1$, where $M1 = \overline{ppw} \cdot pw(\overline{ok} + \overline{rej})$.

The second process $L2$ checks the user id before prompting for the password. If the user id is found correct, it proceeds as above; if not, it will still prompt for the password and reject regardless of the answer. The process $L2$ is thus described as $\overline{pui} \cdot ui(M1 + M2)$, with $M1$ as above and $M2 = \overline{ppw} \cdot pw \cdot \overline{rej}$.

In BPAaa$^{\leq}$, both login processes are equal, because we can demonstrate equality of $M1 + M2$ and $M1$. Since $M1$ is unstable, it follows from $I2$ that $M1 \leq M1 + M2$. It follows from $I1$ that $\overline{ok} + \overline{rej} \geq \overline{rej}$. Closedness under contexts yields $M1 \geq M2$, which implies by $A3$ that $M1 = M1 + M1 \geq M1 + M2$. Thus $M1 = M1 + M2$ by antisymmetry. As a result, $L1 = L2$.

The equality of $L1$ and $L2$ conforms to intuition and is desirable as well, since potential intruders are left uninformed whether they "guessed" a correct user id. At the end of Section 2.3, we will argue that equality of $L1$ and $L2$ cannot be proved without the inequalities $I1$ and $I2$.

2.2 Process Theory

Let $\mathcal{AC}$ be a set of *actions* partitioned into disjoint subsets $\mathcal{A}$ (non-autonomous actions) and $\mathcal{F}$ (autonomous actions). A *process space* is a pair $(\mathcal{P}, \longrightarrow)$, where $\mathcal{P}$ is a set of *processes* and $\bullet \overset{\bullet}{\longrightarrow} \bullet : \mathbb{P}(\mathcal{P} \times \mathcal{AC} \times \mathcal{P} \cup \{\surd\})$ a ternary relation, the *event* relation. The event $p \overset{a}{\longrightarrow} \surd$ represents successful termination of p with action a. A *stable process* cannot perform any autonomous actions in the first step. The set $\mathcal{S}$ of stable processes is thus defined as $\mathcal{S} = \{p : \mathcal{P} \mid \neg \exists f : \mathcal{F};\ p' : \mathcal{P} \cup \{\surd\} \bullet p \overset{f}{\longrightarrow} p'\}$.

We define a class of relations called *partial bisimulations*. A simulation R is a partial bisimulation if R^{-1} is a local simulation for stable processes.

Definition 1 *A relation* $R : \mathbb{P}(\mathcal{P} \cup \{\surd\} \times \mathcal{P} \cup \{\surd\})$ *is called a* partial bisimulation *(PBS) iff it satisfies the following requirements. For* $p, p', q, q' : \mathcal{P} \cup \{\surd\}$ *and* $e : \mathcal{AC}$*,*

$i)$ $pRq \Rightarrow (p = \surd \Leftrightarrow q = \surd) \wedge$
$(pRq \wedge p \xrightarrow{e} p') \Rightarrow \exists q' : \mathcal{P} \bullet (q \xrightarrow{e} q' \wedge p'Rq'),$

$ii)$ $(pRq \wedge p \in \mathcal{S} \wedge q \xrightarrow{e} q') \Rightarrow \exists p' : \mathcal{P} \bullet (p \xrightarrow{e} p' \wedge p'Rq').$

Process $p : \mathcal{P}$ *is said to* implement $q : \mathcal{P}$ *(notation* $p \preceq q$*) iff there exists a PBS* R *such that* pRq*. The processes* p, q *are said to be* equivalent *(notation* $p \simeq q$*) iff* $p \preceq q \wedge q \preceq p$*.*

As usual, we can also define strong bisimulation on processes (see e.g. [BaWe90]). A strong bisimulation between processes p and q is a PBS relating both p to q and q to p. If $\mathcal{F} = \emptyset$, each PBS is a strong bisimulation.

Property 2 *The relation* $\preceq$ *is a preorder and, consequently,* $\simeq$ *is an equivalence relation.*

Proof We have to show that $\preceq$ is reflexive and transitive. The identity relation on processes is a PBS, proving reflexivity. For transitivity, assume that R, S are PBSs such that pRq and qSr. Obviously, the relation composition $R \circ S$ is a simulation. If p is stable, then R^{-1} is a local simulation, so q is stable and hence S^{-1} is a local simulation. So $(R \circ S)^{-1}$ is a local simulation and $R \circ S$ a PBS. So $p \preceq r$. $\square$

With preorder relations like partial bisimulation, special care must be taken with respect to termination and nontermination of processes. It is undesirable that a terminating process can be implemented by a nonterminating one. Therefore, we shall define a termination property for processes. Intuitively, a process is called *terminating* if it has a path leading to $\surd$ and it can only make transitions to terminating processes. A terminating process cannot deadlock, nor can it contain a cycle from which escape is impossible.

Definition 2 *The set* $\mathcal{AC}^*$ *of process* traces *consists of finite sequences* $e_1 \ldots e_n$ *of actions, including the empty trace* ε*. The reachability relation* $\bullet \overset{\bullet}{\Longrightarrow} \bullet : \mathbb{P}((\mathcal{P} \cup \{\surd\}) \times \mathcal{AC}^* \times (\mathcal{P} \cup \{\surd\}))$ *is defined as the smallest relation (ordered by set inclusion) such that for all* $p, p', p'' : \mathcal{P} \cup \{\surd\}$*;* $e : \mathcal{AC}$*;* $\alpha : \mathcal{AC}^*$*,*

$p \overset{\varepsilon}{\Longrightarrow} p \wedge (p \xrightarrow{e} p' \wedge p' \overset{\alpha}{\Longrightarrow} p'' \Rightarrow p \overset{e\alpha}{\Longrightarrow} p'').$

The termination predicate *for a process* p *(notation* $p\Downarrow$*) is defined as follows.*

$p\Downarrow \Leftrightarrow \forall p' : \mathcal{P}; \alpha : \mathcal{AC}^* \bullet (p \overset{\alpha}{\Longrightarrow} p' \Rightarrow \exists \beta : \mathcal{AC}^* \bullet p' \overset{\beta}{\Longrightarrow} \surd).$

2.3 A Process Model for BPAaa$^{\leq}$

In order to construct a process model for BPAaa$^{\leq}$, we have to give interpretations $\mathcal{A}$, $\mathcal{F}$ and $\mathcal{P}$ respectively for the sorts A, F and P. Let the process space $\mathcal{P}$ be the set of *closed* BPAaa$^{\leq}(\mathsf{A})$ terms. Let $\mathcal{A}$ and $\mathcal{F}$ simply be equal to A and F respectively. Obviously, these interpretations do satisfy the requirements $\mathsf{A} \subseteq \mathsf{P}$ and $\mathsf{F} \subseteq \mathsf{P}$. Let $\mathcal{AC} = \mathcal{A} \cup \mathcal{F}$. The relation $\bullet \xrightarrow{\bullet} \bullet$ is given as the smallest relation satisfying the requirements in Table 2. In this table, the variables $a : \mathcal{A}$; $p, p', q : \mathcal{P}$; $e : \mathcal{AC}$ are used. Our model will be the set of equivalence classes of $\mathcal{P}$ modulo PBS equivalence, denoted $\mathcal{P}/\simeq$. The following property means that $\mathcal{P}/\simeq$ with $\preceq$ is a partial order algebra.

Property 3 *The relation* $\preceq$ *is a precongruence w.r.t. the* BPAaa$^{\leq}$ *operators, i.e., for all* $a, b : \mathcal{A}$*,* $a \preceq b$ *implies* $\overline{a} \preceq \overline{b}$ *and for all* $p_1, p_2, q_1, q_2 : \mathcal{P}$*,* $p_1 \preceq p_2$ *and* $q_1 \preceq q_2$ *imply* $p_1 \cdot q_1 \preceq p_2 \cdot q_2$ *and* $p_1 + q_1 \preceq p_2 + q_2$*.*

$$
\begin{array}{|cccc|}
\hline
e \xrightarrow{e} \surd & \dfrac{p \xrightarrow{e} p'}{p \cdot q \xrightarrow{e} p' \cdot q} & \dfrac{p \xrightarrow{e} \surd}{p \cdot q \xrightarrow{e} q} & \\
\dfrac{p \xrightarrow{e} p'}{p + q \xrightarrow{e} p'} & \dfrac{p \xrightarrow{e} p'}{q + p \xrightarrow{e} p'} & \dfrac{p \xrightarrow{e} \surd}{p + q \xrightarrow{e} \surd} & \dfrac{p \xrightarrow{e} \surd}{q + p \xrightarrow{e} \surd} \\
\hline
\end{array}
$$

Table 2: Plotkin-style SOS rules for BPAaa$^{\leq}$

Proof If $a \preceq b$ then $a = b$ and thus $\overline{a} = \overline{b}$. The other two cases are proved by constructing a PBS. □

For closed terms p and q, BPAaa$^{\leq}$ $\vdash p \leq q$ signifies that $p \leq q$ can be derived from the axioms, whereas $\mathcal{P}/\simeq\ \models p \leq q$ signifies that $p \leq q$ is valid in the model $\mathcal{P}/\simeq$. That is, $\mathcal{P}/\simeq\ \models p \leq q$ iff $p \preceq q$. The following theorem states soundness and completeness of BPAaa$^{\leq}$ with respect to $\mathcal{P}/\simeq$.

Theorem 1 *Let p, q be closed* BPAaa$^{\leq}$ *terms. Then* BPAaa$^{\leq}$ $\vdash p \leq q$ *iff* $\mathcal{P}/\simeq\ \models p \leq q$.

Proof It follows from Property 3 that it suffices to check the validity of the axioms to prove soundness. In each case, PBSs can be constructed easily. For the axioms $A1 \ldots A7$, even strong bisimulations can be constructed.

Completeness is more involved. Suppose $p \preceq q$. Then there exists a PBS R such that pRq. We use induction on the number of symbols in p plus the number of symbols in q. The basic case, where p and q each consist of a single symbol follows from the fact that R is a PBS such that pRq. Processes p and q are either both equal to δ or both consist of the same action. Hence, in the basic case, BPAaa$^{\leq}$ $\vdash$ $p \leq q$.

In the induction step, from Property 1, we may assume that p and q are basic terms; so

$$p = \sum_{i:I} a_i + \sum_{j:J} b_j \cdot s_j, \qquad q = \sum_{k:K} c_k + \sum_{l:L} d_l \cdot t_l,$$

where I, J, K and L are finite index sets, the a_i, b_j, c_k and d_l are actions from AC and the s_j and t_l are basic terms. We distinguish two cases.

First, assume that p is stable. Then R^{-1} is a local simulation. We may conclude from the derivation rules in Table 2 that

$$\forall i : I \bullet \exists k : K \bullet a_i = c_k, \qquad \forall j : J \bullet \exists l : L \bullet b_j = d_l \wedge s_j R t_l,$$
$$\forall k : K \bullet \exists i : I \bullet c_k = a_i, \qquad \forall l : L \bullet \exists j : J \bullet d_l = b_j \wedge s_j R t_l.$$

The induction hypothesis yields

$$\forall j : J \bullet \exists l : L \bullet b_j = d_l \wedge \text{BPAaa}^{\leq} \vdash s_j \leq t_l.$$

It follows immediately that BPAaa$^{\leq}$ $\vdash p \leq q$.

Second, assume that p is unstable. Since R is a simulation, we may conclude from the derivation rules in Table 2 that

$$\forall i : I \bullet \exists k : K \bullet a_i = c_k, \quad \forall j : J \bullet \exists l : L \bullet b_j = d_l \wedge s_j R t_l.$$

The induction hypothesis again yields

$$\forall j : J \bullet \exists l : L \bullet b_j = d_l \wedge \text{BPAaa}^{\leq} \vdash s_j \leq t_l.$$

Since p is unstable, one of its initial actions is automonous. Using axioms $I1$ and $I2$, BPAaa$^{\leq}$ $\vdash p \leq q$ follows easily. □

The set of equivalence classes of $\mathcal{P}$ modulo strong bisimulation equivalence is a sound and complete model for BPAaa. This result follows from the one-to-one correspondence between closed BPAaa(A) terms and closed BPA$_\delta$(AC) terms together with soundness and completeness of BPA$_\delta$(AC) for the set of equivalence classes of closed BPA$_\delta$(AC) terms modulo strong bisimulation equivalence. It means that all actions in AC are considered to be non-autonomous and thus all processes are considered to be stable. The consequence is that, as long as no inequality is used, two closed BPAaa$^{\leq}$ terms that are derivably equal are strongly bisimilar. The use of an inequality in a verification represents the reduction of unstable nondeterminism. It is important to keep track at which points this reduction occurs. This can be achieved by delaying the use of $I1$ and $I2$ until they are really needed, i.e., proceeding within BPAaa as long as possible.

At this point, we can also justify our claim that the equivalence of the login procedures $L1$ and $L2$ in the example of Section 2.1 cannot be proved without the inequalities $I1$ and $I2$. This follows immediately from the observations above and the fact that $L1$ and $L2$ are not bisimilar.

3 Iteration

In this section, an iteration operator, c.q. the binary Kleene star (BKS) is added to the basic theory. The additional signature and axioms, which have been taken from [BeBP94], are stated in Table 3.

BPA*aa$^{\leq}$(A)	
BPAaa$^{\leq}$(A)	
$\_ ^* \_ : \mathsf{P} \times \mathsf{P} \to \mathsf{P}$	
$x, y, z : \mathsf{P}$	
$x(x^* y) + y = x^* y$	$BKS1$
$x^*(y \cdot z) = (x^* y)z$	$BKS2$
$x^*(y((x+y)^* z) + z) = (x+y)^* z$	$BKS3$

$$\frac{p \xrightarrow{e} p'}{p^* q \xrightarrow{e} p' \cdot (p^* q), \;\; q^* p \xrightarrow{e} p'} \qquad \frac{p \xrightarrow{e} \surd}{p^* q \xrightarrow{e} p^* q, \;\; q^* p \xrightarrow{e} \surd}$$

Table 3: Axioms and SOS rules for the binary Kleene star

A process model is derived as follows. Again, let $\mathcal{A}$ and $\mathcal{F}$ be equal to A and F. Recall that $\mathcal{AC} = \mathcal{A} \cup \mathcal{F}$. Define $\mathcal{P}$ as the set of closed BPA*aa$^{\leq}$ terms. The relation $\_ \xrightarrow{\_} \_ : \mathbb{P}(\mathcal{P} \times \mathcal{AC} \times \mathcal{P} \cup \{\surd\})$ is defined as the smallest relation satisfying the rules in Tables 2 and 3, where $e : \mathcal{AC}$ and $p, p', q : \mathcal{P}$.

Theorem 2 *Let p, q be closed* BPA*aa$^{\leq}$ *terms. Then* BPA*aa$^{\leq}$ $\vdash p \leq q$ *implies* $\mathcal{P}/\simeq \; \models p \leq q$.

Proof Standard. First, show that $\preceq$ is a precongruence w.r.t. the binary Kleene star. Second, check the validity of $BKS1 \ldots BKS3$ by constructing PBSs. Note that even strong bisimulations can be constructed for these axioms. $\Box$

The following theorem states that the theory BPA*aa$^{\leq}$ does not introduce any inequalities (and therefore equations) between closed BPAaa$^{\leq}$ terms which were not already derivable from BPAaa$^{\leq}$. That is, the theory BPA*aa$^{\leq}$ is a *conservative extension* of BPAaa$^{\leq}$.

Theorem 3 *Let p and q be closed* BPAaa$^{\leq}$ *terms. Then* BPA*aa$^{\leq}$ $\vdash p \leq q$ *iff* BPAaa$^{\leq}$ $\vdash p \leq q$.

Proof (sketch) We use the theory and terminology developed in [D'Ar95]. It follows from the format of the derivation rules that the term deduction system defined by Table 2 is source dependent. Furthermore, it has no rules with negative premises, so it has a unique well supported model. For the same reason, the term deduction system defined by Tables 2 and 3 has a unique well supported model. As a result, it is an *operational* conservative extension (up to $\preceq$) of the term deduction system of Table 2. It then follows from Theorems 1 and 2 that BPA*aa$^{\leq}$ is a conservative extension of BPAaa$^{\leq}$. $\Box$

A complete finite axiomatization of BPA$^*_\delta$ and thus BPA*aa$^{\leq}$ is impossible ([Sew94]; c.f. [FoZa94] for related results).

For reasoning with iterative processes, we formulate some conditional equations and inequalities. RSP^* is the iteration variant of the "Recursive Specification Principle" (c.f. e.g. [BaWe90]); the other names are derived from "Iteration Inequality." For any $v, w, x, y, z : \mathcal{P}$,

$$\frac{v \cdot x + w \leq x}{v^* w \leq x}\ IIL \qquad \frac{x \leq y \cdot x + z}{x \leq y^* z}\ IIR \qquad \frac{x = y \cdot x + z}{x = y^* z}\ RSP^*$$

$$\frac{v \cdot x + w \leq x \leq y \cdot x + z,\ w \leq z}{v^* w \leq x \leq y^* z}\ II$$

Clearly, $(IIL \wedge IIR) \Rightarrow II \Rightarrow RSP^*$. It is not hard to show that RSP^* and $BKS1$ imply $BKS2$ and $BKS3$.

Theorem 4 *The above derivation rules are valid in the model $\mathcal{P}/\simeq$.*

Proof (sketch) Inequality IIR is proved by constructing a PBS as follows. Let p, q and r be closed BPA*aa$^\leq$ terms and let R be a PBS such that $pR(q \cdot p + r)$. R^k is the k-fold composition of R, where R^0 equals the identity relation and R^1 equals R. The relation Q is defined as the smallest relation such that $\surd Q \surd$ and for any $k \geq 0$, closed term ξ and open term $E(X)$ with X as its only variable, $\xi Q \xi$ and $\xi R^k E(p) \Rightarrow \xi Q E(q^* r)$. Using the operational semantics, it can be shown that this relation is a PBS relating p and $q^* r$. So IIR is valid. The validity of IIL is shown using a symmetrical argument. $\Box$

We have thus shown the validity of IIL, IIR, II and RSP^*. However, the inequalities IIL and IIR permit the derivation of, e.g., $\overline{a}^*\delta \leq \overline{a}^* b$. This is undesirable, since the second process is terminating, whereas the first process is not. For that reason, in the remainder, we only use the weaker inequality II. In BPA*aa$^\leq + II$, a terminating process cannot be implemented by a nonterminating process.

Property 4 *Let $p, q : \mathcal{P}$. If* BPA*aa$^\leq + II \vdash p \leq q$ *and $q \Downarrow$, then $p \Downarrow$.*

Proof (sketch) Structural induction on p is used. The basic case is trivial. In the induction step, we consider three cases. First, if $p = p_1 + p_2 \leq q$, we may assume that $\delta \neq p_1 \neq p_2 \neq \delta$. The only way that $p \leq q$ can be deduced from BPA*aa$^\leq + II$ is when q has the form $q_1 + q_2$ and BPA*aa$^\leq + II \vdash (p_1 \leq q_1 \wedge p_2 \leq q_2)$. If $q \Downarrow$, then $q_1 \Downarrow$ and $q_2 \Downarrow$, allowing completion of the induction step in this case. Second, if $p = p_1 \cdot p_2$, the argument is similar. Third, if $p = p_1 {}^* p_2$, $p \leq q$ can be derived in two ways.

The first possibility is that q has the form $q_1 {}^* q_2$ and BPA*aa$^\leq + II \vdash (p_1 \leq q_1 \wedge p_2 \leq q_2)$, which is dealt with as above. The second possibility follows from II if $p_1 \cdot q + p_2 \leq q \leq r \cdot q + s$ and $p_2 \leq s$, for some processes r and s. Now suppose $q \Downarrow$. We have to show that $p \Downarrow$, i.e. that $p_1 \Downarrow$ and $p_2 \Downarrow$. Assume $p_2 = \delta$. Then $s = \delta$ and $q \leq r \cdot q$. Soundness gives $q \preceq r \cdot q$. Since $q \Downarrow$, it follows that $q \stackrel{\alpha}{\Longrightarrow} \surd$ for some trace α. So there exist nonempty traces β, γ such that $\alpha = \beta\gamma$, $r \stackrel{\beta}{\Longrightarrow} \surd$ and $q \stackrel{\gamma}{\Longrightarrow} \surd$. So for any path of q leading to $\surd$, a shorter path can be found, which is a contradiction. So $p_2 \neq \delta$. Hence, one deduces again that q has the form $q_1 + q_2$ and BPA*aa$^\leq + II \vdash (p_1 \cdot q \leq q_1 \wedge p_2 \leq q_2)$, which can again be treated as above. $\Box$

4 Concurrency and Communication

Table 4 gives the additional operators and axioms for the extension of ACP with autonomous actions. An extra parameter, the communication function $\gamma : \mathsf{A} \cup \{\delta\} \times \mathsf{A} \cup \{\delta\} \rightarrow \mathsf{A} \cup \{\delta\}$, is added, that is commutative and associative and satisfies $\gamma(a, \delta) = \delta$ for all atoms a.

The UCM axioms disallow synchronization of unstable processes. The SCM axioms are $CM8$ and $CM9$ of ACP restricted to stable processes. The other merge-related axioms are standard.

The auxiliary predicate $S(\centerdot)$ stating that a process is stable is defined as follows. For $e : \mathsf{AC}$; $x, y : \mathsf{P}$,

$$S(\delta) \qquad S(e) \Leftrightarrow e \in \mathsf{A}$$
$$S(x \cdot y) \Leftrightarrow S(x) \qquad S(x + y) \Leftrightarrow S(x) \wedge S(y)$$

Only a few of the other operators need explanation. Encapsulation of autonomous actions is impossible ($\overline{D}$), which is the essence of being autonomous, as explained in the introduction. The renaming operator t_I renames the atoms from a given set I into a special atom t; autonomized atoms from I are renamed into $\overline{t}$. An operator aut_I is added, that autonomizes the atoms from I. The composed operator $t_I \circ aut_I$ has the properties of pre-abstraction [BaBe88].

ACPaa$^{\leq}(\mathsf{A}, \gamma)$

BPAaa$^{\leq}(\mathsf{A})$

$t : \mathsf{A} \quad \partial_{\centerdot}(\centerdot),\ aut_{\centerdot}(\centerdot),\ t_{\centerdot}(\centerdot) : \mathbb{P}(\mathsf{A}) \times \mathsf{P} \to \mathsf{P} \quad \centerdot \parallel \centerdot,\ \centerdot \mathbin{\|\!\_} \centerdot,\ \centerdot \mid \centerdot : \mathsf{P} \times \mathsf{P} \to \mathsf{P}$

$a : \mathsf{A}$; $b, c : \mathsf{A} \cup \{\delta\}$; $e : \mathsf{AC} \cup \{\delta\}$; $x, y, z : \mathsf{P}$; $H, I : \mathbb{P}(\mathsf{A})$; $L : \{\partial_H, aut_I, t_I\}$

$a \notin H \Rightarrow \partial_H(a) = a$	$D1$	$b \mid c = \gamma(b, c)$	CF
$a \in H \Rightarrow \partial_H(a) = \delta$	$D2$		
$\partial_H(\overline{a}) = \overline{a}$	$\overline{D}$	$x \parallel y = x \mathbin{\|\!\_} y + y \mathbin{\|\!\_} x + x \mid y$	$CM1$
		$e \mathbin{\|\!\_} x = e \cdot x$	$CM2$
$a \notin I \Rightarrow aut_I(a) = a$	$AU1$	$e \cdot x \mathbin{\|\!\_} y = e \cdot (x \parallel y)$	$CM3$
$a \in I \Rightarrow aut_I(a) = \overline{a}$	$AU2$	$(x + y) \mathbin{\|\!\_} z = x \mathbin{\|\!\_} z + y \mathbin{\|\!\_} z$	$CM4$
$aut_I(\overline{a}) = \overline{a}$	$AU3$	$b \cdot x \mid c = (b \mid c)x$	$CM5$
		$b \mid c \cdot x = (b \mid c)x$	$CM6$
$a \notin I \Rightarrow t_I(a) = a$	$RN1$	$b \cdot x \mid c \cdot y = (b \mid c)(x \parallel y)$	$CM7$
$a \in I \Rightarrow t_I(a) = t$	$RN2$	$S(x + y) \Rightarrow (x + y) \mid z = x \mid z + y \mid z$	$SCM8$
$t_I(\overline{a}) = \overline{t_I(a)}$	$\overline{RN}$	$S(y + z) \Rightarrow x \mid (y + z) = x \mid y + x \mid z$	$SCM9$
		$(\overline{a} + x) \mid y = \delta$	$UCM1$
$L(\delta) = \delta$	$L1$	$(\overline{a} \cdot x + y) \mid z = \delta$	$UCM2$
$L(x + y) = L(x) + L(y)$	$L2$	$x \mid (\overline{a} + y) = \delta$	$UCM3$
$L(x \cdot y) = L(x) \cdot L(y)$	$L3$	$x \mid (\overline{a} \cdot y + z) = \delta$	$UCM4$

Table 4: ACP with autonomous actions

The following theorem shows that the non-BPAaa$^{\leq}$ operators can be eliminated.

Theorem 5 *For every closed* ACPaa$^{\leq}$ *term p, there is a basic* BPAaa$^{\leq}$ *term s such that* ACPaa$^{\leq}$ $\vdash$ $p = s$.

Proof (sketch) Consider ACPaa$^{\leq}$ as a term rewriting system. That is, consider the axioms $A3 \ldots A7$ plus all the axioms in Table 4 as rewrite rules from left to right. We can use the method of recursive path ordering to show that this term rewriting system is strongly normalizing (see [BaVe95] for a detailed description). Here, we only give the partial ordering on the ranked operators of ACPaa$^{\leq}$. For $k > 1$ and $a : \mathsf{A}$,

$$\delta < a \quad \{a, \bar{\cdot}, +\} < \cdot < \{\mathbin{\|\!\_}_2, |_2, \partial_H, aut_I, t_I\} \quad \{\mathbin{\|\!\_}_k, |_k\} < \|_k < \{\mathbin{\|\!\_}_{k+1}, |_{k+1}\}$$

Now, it can be verified by structural induction that the normal forms are basic BPAaa$^{\leq}$ terms. The basic step is simple. For the induction step, note that for any normal form containing a non-BPAaa$^{\leq}$ operator applied to normal forms, which by the induction hypothesis are basic BPAaa$^{\leq}$ terms, a rewrite rule can be found in Table 4, which is a contradiction. $\square$

A process model is defined as before. The process space $\mathcal{P}$ is defined as the set of closed ACPaa$^{\leq}$ terms. The relation $\boldsymbol{.} \stackrel{\bullet}{\longrightarrow} \boldsymbol{.}$ is the smallest relation satisfying the rules in Tables 2 and 5. Recall that in Section 2.2, $\mathcal{S}$ is defined as the set of stable processes. Note that for all $p : \mathcal{P}$, $p \in \mathcal{S} \Leftrightarrow S(p)$. Let $a, b, c : \mathcal{A}$; $e : \mathcal{AC}$; $H, I : \mathbb{P}(\mathcal{A})$; $p, p', q, q' : \mathcal{P}$ and $\hat{p}, \hat{q} : \mathcal{S}$.

$$\frac{p \xrightarrow{e} p'}{p \| q \xrightarrow{e} p' \| q,\ q \| p \xrightarrow{e} q \| p',\ p \lfloor\!\lfloor q \xrightarrow{e} p' \| q} \qquad \frac{p \xrightarrow{e} \surd}{p \| q \xrightarrow{e} q,\ q \| p \xrightarrow{e} q,\ p \lfloor\!\lfloor q \xrightarrow{e} q}$$

$$\frac{\hat{p} \xrightarrow{a} p', \hat{q} \xrightarrow{b} q', \gamma(a,b) = c}{\hat{p} \mid \hat{q} \xrightarrow{c} p' \| q', \hat{p} \| \hat{q} \xrightarrow{c} p' \| q'} \qquad \frac{\hat{p} \xrightarrow{a} \surd, \hat{q} \xrightarrow{b} \surd, \gamma(a,b) = c}{\hat{p} \mid \hat{q} \xrightarrow{c} \surd, \hat{p} \| \hat{q} \xrightarrow{c} \surd}$$

$$\frac{\hat{p} \xrightarrow{a} p', \hat{q} \xrightarrow{b} \surd, \gamma(a,b) = c}{\hat{p} \mid \hat{q} \xrightarrow{c} p', \hat{q} \mid \hat{p} \xrightarrow{c} p', \hat{p} \| \hat{q} \xrightarrow{c} p', \hat{q} \| \hat{p} \xrightarrow{c} p'}$$

$$\frac{p \xrightarrow{a} p',\ a \notin H}{\partial_H(p) \xrightarrow{a} \partial_H(p')} \qquad \frac{p \xrightarrow{a} \surd,\ a \notin H}{\partial_H(p) \xrightarrow{a} \surd}$$

$$\frac{p \xrightarrow{a} p',\ a \notin I}{aut_I(p) \xrightarrow{a} aut_I(p'),\ t_I(p) \xrightarrow{a} t_I(p')} \qquad \frac{p \xrightarrow{a} \surd,\ a \notin I}{aut_I(p) \xrightarrow{a} \surd,\ t_I(p) \xrightarrow{a} \surd}$$

$$\frac{p \xrightarrow{a} p',\ a \in I}{aut_I(p) \xrightarrow{\overline{a}} aut_I(p'),\ t_I(p) \xrightarrow{t} t_I(p')} \qquad \frac{p \xrightarrow{a} \surd,\ a \in I}{aut_I(p) \xrightarrow{\overline{a}} \surd,\ t_I(p) \xrightarrow{t} \surd}$$

$$\frac{p \xrightarrow{\overline{a}} p'}{\partial_H(p) \xrightarrow{\overline{a}} \partial_H(p'),\ aut_I(p) \xrightarrow{\overline{a}} aut_I(p')} \qquad \frac{p \xrightarrow{\overline{a}} \surd}{\partial_H(p) \xrightarrow{\overline{a}} \surd,\ aut_I(p) \xrightarrow{\overline{a}} \surd}$$

$$\frac{p \xrightarrow{\overline{a}} p',\ a \notin I}{t_I(p) \xrightarrow{\overline{a}} t_I(p')} \qquad \frac{p \xrightarrow{\overline{a}} p',\ a \in I}{t_I(p) \xrightarrow{\overline{t}} t_I(p')} \qquad \frac{p \xrightarrow{\overline{a}} \surd,\ a \notin I}{t_I(p) \xrightarrow{\overline{a}} \surd} \qquad \frac{p \xrightarrow{\overline{a}} \surd,\ a \in I}{t_I(p) \xrightarrow{\overline{t}} \surd}$$

Table 5: SOS rules for non-BPAaa$^{\leq}$ operators

The following property and theorem deal with soundness of ACPaa$^{\leq}$.

Property 5 *The $\preceq$ preorder is a precongruence w.r.t. the* ACPaa$^{\leq}$ *operators.*

Proof Let R_1 and R_2 be PBSs. Let $Q = \bigcup p_1, p_2, q_1, q_2 : \mathcal{P} \mid p_1 R_1 q_1 \wedge p_2 R_2 q_2 \bullet \{(p_1 \| p_2, q_1 \| q_2)\}$. It is easy to check that $Q \cup R_1 \cup R_2$ is a PBS, noticing that $x \| y$ is stable iff both x and y are stable. So for any $p_1, p_2, q_1, q_2 : \mathcal{P}$, $(p_1 \preceq q_1 \wedge p_2 \preceq q_2) \Rightarrow p_1 \| p_2 \preceq q_1 \| q_2$. So $\preceq$ is a precongruence for the merge operator. For the left merge operator, under the same assumptions, $Q \cup R1 \cup R2 \cup \bigcup p_1, p_2, q_1, q_2 : \mathcal{P} \mid p_1 R_1 q_1 \wedge p_2 R_2 q_2 \bullet \{(p_1 \lfloor\!\lfloor p_2, q_1 \lfloor\!\lfloor q_2)\}$ is a PBS; a similar PBS is constructed for the communication merge. In the latter case it is essential that communication with unstable processes is δ, so the autonomous actions of an unstable process cannot be selectively blocked.

Let R be a PBS. Define $Q = \{p, q : \mathcal{P} \mid pRq \bullet (\partial_H(p), \partial_H(q))\}$. It is easy to show that $Q \cup \{(\surd, \surd)\}$ is a PBS, whence $\preceq$ is a precongruence for encapsulation. Like before, it is essential that the encapsulation of an autonomous action is not δ. The t and aut operators are treated likewise. $\square$

We can now prove soundness and completeness. For Theorem 6 it suffices to check the validity of all the axioms. In each case it is straightforward to give a strong bisimulation for the added axioms. Theorem 7 is proved in the same way as Theorem 3. After that, completeness follows from Theorems 1, 5, 6 and 7. See [D'Ar95] for more details.

Theorem 6 *Let p and q be closed* ACPaa$^{\leq}$ *terms. Then* ACPaa$^{\leq}$ $\vdash p \leq q$ *implies* $\mathcal{P}/\simeq\, \models p \leq q$.

Theorem 7 *Let p and q be closed* BPAaa$^{\leq}$ *terms. Then* ACPaa$^{\leq}$ $\vdash p \leq q$ *iff* BPAaa$^{\leq}$ $\vdash p \leq q$.

Theorem 8 *Let p and q be closed* ACPaa$^{\leq}$ *terms. Then* ACPaa$^{\leq}$ $\vdash p \leq q$ *iff* $\mathcal{P}/\simeq\, \models p \leq q$.

5 Example: ABP

The benchmark example for process algebra is the Alternating Bit Protocol (ABP). As the ABP has recursion as well as communication, we define the theory ACP*aa$^{\leq}$ by combining ACPaa$^{\leq}$ and BPA*aa$^{\leq}$, adding the axiom $L4$:

$L(x * y) = L(x) * L(y)$ for $H, I : \mathbb{P}(\mathsf{A})$; $L : \{\partial_H, aut_I, t_I\}$; $x, y : \mathsf{P}$.

A process model can be defined as before. Soundness of ACP*aa$^{\leq}$ follows from Theorems 2 and 6 and the validity of axiom $L4$. ACP*aa$^{\leq}$ is a conservative extension of ACPaa$^{\leq}$ (and thus of BPAaa$^{\leq}$). A finite complete axiomatization does not exist. Inequality II and hence equation RSP^* are still valid. Furthermore, Property 4 can be generalized to closed ACP*aa$^{\leq}$ terms.

A protocol specification usually is of the form $(\sum_{d:D} in_d \cdot out_d) * \delta$, a never-ending loop of reading a message at one end and producing the same message at the other end. Verifications based on abstraction give an implementation with internal actions renamed to τ and use the special properties of the silent action τ to prove the implementation equal to the specification. Use is made of a "fairness" axiom, a simple variant of which can be stated as $x(\tau * y) = x \cdot y$. Although such a fairness axiom is sound in many bisimulation-oriented models, one may disagree with it and use more discriminating divergence-sensitive models instead, in which case the specification equals $(\sum_{d:D} in_d(\tau * \tau) out_d(\tau * \tau)) * \delta$.

In this paper the latter approach is followed, but without silent actions and using pre-abstraction instead. Internal actions are renamed into an autonomous, but visible internal action $\bar{t}$. Our specification ABP_{spec} now reads $(\sum_{d:D} in_d(\bar{t}*\bar{t}) out_d(\bar{t}*\bar{t}))*\delta$, i.e., before any external action, a *terminating* $\bar{t}$-loop may take place. ABP_{spec} requires at least one $\bar{t}$-step to be made between two external actions; a more complex specification can be given that also allows external actions to succeed one another without $\bar{t}$ steps in between. For the sake of simplicity we use ABP_{spec} as given above.

An implementation ABP_{imp} must satisfy $ABP_{imp} \leq ABP_{spec}$. A proper implementation never loses the option of proceeding to a next external action, so it should be of the form $CYC * \delta$ with $CYC \leq \sum_{d:D} in_d(\bar{t} * \bar{t}) out_d(\bar{t} * \bar{t})$ and $CYC \Downarrow$. It follows from the (generalized) Property 4 that this can be established by only using the axioms from ACP*aa$^{\leq}$ + II in the verification and avoiding the use of IIL and IIR.

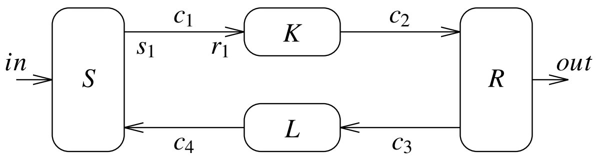

Figure 1: ABP implementation.

In general, an implementation takes the form $t_I \circ aut_J \circ \partial_H(\|_{k:K} P_k)$, where I, J are sets of actions internal to the implementation, H the actions that are enforced to communicate and P_k for k in K the component processes. The actions in $J \setminus I$, while remaining internal, are still visible from outside.

If such actions are not deemed necessary, I and J can be taken equal, as in the ABP implementation below, that has been adapted from [BeKl86]. It consists of a sender S, a receiver R, a message channel K and an acknowledgement channel L. See Figure 1. Let D be the finite set of data to be transmitted. For acknowledgements, the element $ack \notin D$ is added and for perturbed communications the element $\perp \notin D \cup \{ack\}$. The set $B = \{0, 1\}$ contains the alternating bit. For $b : B$, its negation $1 - b$ is written b'. We define the set M of messages as $((D \cup \{ack\}) \times B) \cup \{\perp\}$. For $d : D \cup \{ack\}; b : B$, we denote the message (d, b) by d_b. The set $C = \{1, 2, 3, 4\}$ defines the communication ports. We add the rules for standard read/send communication. For $k : C; m : M, \gamma(r_k(m), s_k(m)) = c_k(m)$ (and its commutative variant). All other communications are set to δ. Let $H = \bigcup k : C; m : M \bullet \{r_k(m), s_k(m)\}$ be the set of communication actions and $I = \{k : C; m : M \bullet c_k(m)\}$ the set of internal actions. Now we can define ABP_{imp} as follows. For $b \in B$, $ABP_{imp} = t_I \circ aut_I \circ \partial_H(S \parallel K \parallel L \parallel R)$, where

$S = (S_0 \cdot S_1) * \delta \qquad S_b = \sum_{d:D} in_d \cdot s_1(d_b)(((r_4(ack_{b'}) + r_4(\perp))s_1(d_b)) * r_4(ack_b))$

$R = (R_0 \cdot R_1) * \delta \qquad R_b = ((\sum_{d:D} r_2(d_{b'}) + r_2(\perp))s_3(ack_{b'})) * ((\sum_{d:D} r_2(d_b) out_d) s_3(ack_b))$

$K = P_{1,2} \quad L = P_{3,4} \qquad P_{i,j} = (\sum_{m:M} r_i(m)(\overline{t} \cdot s_j(m) + \overline{t} \cdot s_j(\perp))) * \delta$

By standard techniques and RSP^*, one derives

$ABP_{imp} = (\sum_{d:D} in_d \cdot \overline{t}(\overline{t}^6 * \overline{t}^2) out_d \cdot \overline{t}(\overline{t}^6 * \overline{t}^2)) * \delta.$

Here, $\overline{t}^k$ is the abbreviation of $\overline{t} \cdot \overline{t} \ldots \overline{t}$ iterated k times. The fact that $ABP_{imp} \leq ABP_{spec}$ follows from the following derivation, showing that $\overline{t}^k(\overline{t}^l * \overline{t}^m) \leq \overline{t} * \overline{t}$ for any $k, l, m > 0$.

By $BKS1$, $\overline{t} * \overline{t} = \overline{t}(\overline{t} * \overline{t}) + \overline{t}$. So from $I1$ and $I2$,

$\overline{t} \leq \overline{t} * \overline{t}, \quad \overline{t}(\overline{t} * \overline{t}) \leq \overline{t} * \overline{t}.$

By induction, and the transitivity of $\leq$, we conclude that for all $k > 0$,

$\overline{t}^k(\overline{t} * \overline{t}) \leq \overline{t} * \overline{t}, \quad qt^l \leq \overline{t} * \overline{t}.$

Using these results, $A3$, $BKS1$ and $I1$, we deduce that for any $l, m > 0$,

$\overline{t}^l(\overline{t} * \overline{t}) + \overline{t}^m \leq \overline{t} * \overline{t} + \overline{t} * \overline{t} = \overline{t} * \overline{t} = \overline{t}(\overline{t} * \overline{t}) + \overline{t} \leq \overline{t}(\overline{t} * \overline{t}) + \overline{t} + \overline{t}^m.$

Applying II, we obtain $\overline{t}^l * \overline{t}^m \leq \overline{t} * \overline{t}$. This result, the fact that for all $k > 0$, $\overline{t}^k(\overline{t} * \overline{t}) \leq \overline{t} * \overline{t}$ and the transitivity of $\leq$ yield

$\overline{t}^k(\overline{t}^l * \overline{t}^m) \leq \overline{t} * \overline{t}.$

This concludes our verification. Since the verfication took place within ACP*aa$^{\leq}$ + II, the implementation ABP_{imp} is a proper one.

6 Conclusions and Further Work

Actions with special properties do enrich process algebra. The present paper discusses autonomous actions, as an alternative to the "partial choice" operator of [BaBe94]. This leads to an algebra with similar expressive power and simpler axioms and models. Terms in the theory $\mathsf{BPA}_{\uplus}$, Basic Process Algebra with partial choice, can be expressed in BPAaa$^{\leq}$ by means of a single autonomous action $\overline{t}$ and following the p-graph construction of [BaBe94]. For example, the expression $(a \uplus b) + c$ is mapped to $\overline{t}(a + c) + \overline{t}(b + c)$.

For reasoning with infinite processes, the II, IIL and IIR inequalities have been introduced. These inequalities may be carried over to other simulation-oriented preorders and can probably be widened in scope to (guarded) recursion.

One can imagine other kinds of special actions, like *delayable* actions $\tilde{a}$ or *irrevocable* actions $\hat{a}$ that e.g. satisfy the equations $\tilde{a} \cdot x = \tilde{a} \| x$ (c.f. the silent step in weak bisimulation) and $x \cdot \hat{a} \cdot y + x \cdot \hat{a} \cdot z = x(\hat{a} \cdot y + \hat{a} \cdot z)$ (c.f. ready trace semantics, [Gla90]). Another extension is the addition of the silent action τ. Branching bisimulation [GlWe89] can be extended to partial branching bisimulation along the lines of this paper, giving an algebra with two different silent actions: τ and $\overline{\tau}$. Both satisfy the branching equations; only the latter satifies $I1$ and $I2$. Although the ABP example shows that silent actions and full abstraction have ceased to be necessary for some verifications, they nevertheless simplify matters considerably.

The theory in this paper can be applied to Petri Nets like in [BaVo95] to discriminate between the consumption and production of tokens. Production of tokens can be considered as autonomous in the sense of the present paper. This leads to an inequational theory for nets with similar advantages with respect to software engineering as stated in the introduction.

References

Abr87 S. Abramsky: *Observation Equivalence as a Testing Equivalence*, TCS 53, pp. 225-241 (1987)

BaBe88 J.C.M. Baeten, J.A. Bergstra: *Global Renaming Operators in Concrete Process Algebra*, I&C 78, pp. 205-245 (1988)

BaBe94 J.C.M. Baeten, J.A. Bergstra: *Process Algebra with Partial Choice*, in: Johnson, Parrow (eds.) *Proc. CONCUR 94*, pp. 465-480, LNCS 836, Springer 1994

BaVe95 J.C.M. Baeten, C. Verhoef: *Concrete Process Algebra*, in: Abramsky, Gabbay, Maibaum (eds.) *Handbook of Logic in Computer Science*, Vol. 4, pp. 149-268, Oxford Univ. Press 1995

BaWe90 J.C.M. Baeten, W.P. Weijland: *Process Algebra*, Cambridge University Press 1990

BaVo95 T. Basten, M. Voorhoeve: *An Algebraic Semantics for Hierarchical P/T Nets (extended abstract)*, in: *Proc. Applications and Theory of Petri Nets*, pp. 45-65, LNCS 935, Springer 1995

BeBP94 J.A. Bergstra, I. Bethke, A. Ponse: *Process Algebra with Iteration and Nesting*, The Computer Journal 37(4), pp. 241-258 (1994)

BeKl86 J.A. Bergstra, J. Klop: *Verification of an alternating bit protocol by means of process algebra* in: W. Bibel, K.P. Jantke: *Math. Methods of Spec. and Synth. of Softw. Systems*, pp. 9-23, LNCS 215, Springer 1986

D'Ar95 P.R. D'Argenio: *A General Conservative Extension Theorem in Process Algebras with Inequalities*, in: Ponse, Verhoef, v. Vlijmen (eds.) *Proc. ACP95*, Computing Science Reports 95-14, pp. 67-79, Eindhoven Univ. Tech. 1995.

FoZa94 W.J. Fokkink, H. Zantema: *Basic Process Algebra with Iteration: Completeness of its Equational Axioms*, The Computer Journal 37(4), pp. 259-267 (1994)

GlWe89 R.J. van Glabbeek, W.P. Weijland: *Branching Time and Abstraction in Bisimulation Semantics*, in: G.X. Ritter (ed.): *Information Processing : Proc. IFIP 11th World Computer Congress*, pp. 613-618, North-Holland 1989

Gla90 R.J. van Glabbeek: *The Linear Time - Branching Time Spectrum*, in: Baeten, Klop (eds.) *Proc. CONCUR 90*, pp. 278-297, LNCS 458, Springer 1990

Hen88 M. Hennessy: *Algebraic Theory of Processes*, MIT Press 1988

Hoa85 C.A.R. Hoare: *Communicating Sequential Processes*, Prentice-Hall 1985

Sew94 P. Sewell: *Bisimulation is not (First Order) Equationally Axiomatizable*, in: Proceedings LICS 1994, pp. 62-70, IEEE Comp. Soc. Press, 1994

Wal90 D.J. Walker: *Bisimulation and Divergence*, I&C 85, pp. 202-241 (1990)