University of Amsterdam
Programming Research Group

Metric Denotational Semantics for BPPA

T.D. Vu

Report PRG0503 July 2005

T.D. Vu

Programming Research Group
Faculty of Science
University of Amsterdam

Kruislaan 403
1098 SJ Amsterdam
The Netherlands

tel. +31 20 525.7585
e-mail: tdvu@science.uva.nl

Programming Research Group Electronic Report Series

Metric denotational semantics for BPPA

Thuy Duong Vu

Programming Research Group, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam, the Netherlands
tdvu@science.uva.nl

Abstract. Program algebra (PGA) is a basic and simple concept of a programming language which has been formulated by Bergstra and Loots in [9, 10]. Behaviors for programs in PGA can be given in the Basic Polarized Process Algebra (BPPA). Based on the theory of metric spaces as introduced in [5], we give a denotational semantics for BPPA. Models of BPPA are considered as complete metric spaces of a suitable mathematical structure. We show that a space consisting of projective sequences is an appropriate model for BPPA. Furthermore, using Banach's fixed point theorem, we prove that the specification of a regular process in this space has a unique solution. This result suggests a model consisting of regular processes. We complete the paper by comparing several models of BPPA.

1 Introduction

In [9, 10], Bergstra and Loots introduce the basic program algebra PGA (ProGram Algebra) as an algebraic framework for the study of sequential programming. The primary application of PGA can be found in teaching because of its simplicity and easy memorizability. The virtue of PGA is that it can be used to answer the question "What is a programming language?" by providing simple and general constructions. The reader is referred to [9, 10] for details.

The description of processes is necessary to give a formal semantics for PGA. This description is based on Basic Polarized Process Algebra (BPPA) which maps programs into behaviors (or processes) [10]. Finding a model for BPPA is important because it helps to explain unexpected behaviors of programs. Moreover, it can help visualize processes. We shall employ the methodology of *denotational* semantics [18, 16] to approach this issue, whereby processes are defined as elements of some suitable mathematical structure.

In this paper, we will illustrate the technique taken from metric topology as introduced in [5, 1, 3] to give a denotational semantics for BPPA. We show that a metric space consisting of projective sequences is an appropriate model for BPPA. Furthermore, we prove that the specification of a regular process determines a unique process by means of Banach's fixed point theorem. This suggests a model consisting of regular processes. We also show that the completion technique will give the same result as the approach based on complete partial orders (cpo's) [6]. Finally, we discuss extensions of some models with *abstraction* [2] which is an important operator in process algebra.

The structure of this paper is as follows. Section 2 introduces the syntax of PGA and the description of BPPA. Section 3 shows that BPPA can be modeled as complete partial orders. In Section 4 we define models of BPPA as complete metric spaces. In Section 5 we deal with the uniqueness of the solution of a regular process specification. Section 6 compares the various models of BPPA. The paper is ended with some concluding remarks in Section 7.

2 PGA and BPPA

In this section, we recall the concepts of PGA and the description of program behaviors based on BPPA from [9, 10] .

2.1 The syntax of PGA

Let Σ be a set of *basic instructions*. Each basic instruction returns a boolean value upon execution.

Definition 1. *The collection of program objects in* PGA *over* Σ*, denoted by* PGA_Σ*, is generated by* **primitive instructions** *and two composition constructs. These primitive instructions are defined by*

- **Basic instruction**. *All* $a \in \Sigma$ *are basic instructions. By the execution of a basic instruction, a boolean value is generated and a state may be modified. After execution, a program has to execute its subsequent instruction. If that instruction fails to exist, inaction occurs.*
- **Termination instruction***, denoted by* $!$*, indicates termination of the program. It does not modify the state and does not return a boolean value.*
- **Positive test instruction**. *For each* $a \in \Sigma$*, there is a positive test instruction denoted by* $+a$*. If* $+a$ *is performed by a program, the state is affected according to* a*. In case true is returned, the subsequent instruction is performed. If there is no remaining instruction, inaction occurs. In case* $false$ *is returned, the next instruction is skipped and the execution continues with the following instruction. If no such instruction exists, inaction occurs.*
- **Negative test instruction**. *For each* $a \in \Sigma$*, there also exists the negative test instruction denoted by* $-a$*. If* $-a$ *is performed by a program, the state is affected according to* a*. In case* $false$ *is returned, the subsequent instruction is executed. If there is no subsequent instruction, inaction occurs. In case true is returned, the next instruction is skipped and the execution proceeds with the following instruction. If no such instruction exists, inaction occurs.*
- **Forward jump instruction**. *For any natural number* k*, there is an instruction* $\#k$ *which denotes a jump of length* k*. The number* k *is the counter of the jump instruction.*
  - *If* $k = 0$*, the jump is to itself (zero steps forward). In this case inaction will result.*
  - *If* $k = 1$*, the instruction is skipped. The subsequent instruction will be executed next. If there is no such instruction, inaction will occur.*

- *If $k > 1$, the execution will skip itself and the next $k-1$ instructions. The instruction after that will be performed. If there is no such instruction, inaction will occur.*

The two composition constructs are defined by

- **Concatenation**. *The concatenation of two programs X and Y in* PGA_Σ*, denoted by $X;Y$, is also in* PGA_Σ*.*
- **Repetition**. *The repetition of a program X in* PGA_Σ*, denoted by X^ω, is also in* PGA_Σ*.*

Table 1 presents program object equations that identify representations of the same single pass instruction sequences (1-4), and structural congruence equations that take care of the simplification of chained jumps (5-8). Here $X^1 = X$,

Table 1. Program object equations and structural congruence equations

$(X;Y);Z{=}X;(Y;Z)$	(1)
$(X^n)^\omega{=}X^\omega$	(2)
$X^\omega;Y{=}X^\omega$	(3)
$(X;Y)^\omega{=}X;(Y;X)^\omega$	(4)
$\#n+1;u_1;\ldots;u_n;\#0{=}\#0;u_1;\ldots;u_n;\#0$	(5)
$\#n+1;u_1;\ldots;u_n;\#m{=}\#n+m+1;u_1;\ldots;u_n;\#m$	(6)
$(\#n+k+1;u_1;\ldots;u_n)^\omega{=}(\#k;u_1;\ldots;u_n)^\omega$	(7)
$X = u_1;\ldots;u_n;(v_1;\ldots;v_{m+1})^\omega \rightarrow \#n+m+k+2;X = \#n+k+1;X$	(8)

$X^{n+1} = X;X^n$, n is a positive integer.

By these program object equations, the unfolding identity of repetition can be obtained: $X^\omega = X;X^\omega$.

Definition 2. *A program object is* **finite** *if it does not contain repetition, otherwise it is* **infinite**.

Example 1. Examples of program objects in PGA are

1. $X = +a;\#3;b;!;c$,
2. $Y = (a;b;\#2)^\omega$.

In the next section, the behaviors of these programs are determined by means of BPPA.

2.2 Primitives of BPPA

The basic instructions in Σ are now also called actions.

Definition 3. BPPA *is defined with the following meaning*

– **Termination**, *denoted by* S*, yields the terminating behavior.*
– **Inactive behavior**, *denoted by* D*, represents the inaction behavior.*
– **Postconditional composition**: *The process* $P \trianglelefteq a \trianglerighteq Q$*, where* $a \in \Sigma$*, first performs* a *and then proceeds with* P *if true was returned or with* Q *otherwise.*
– **Action prefix**: *For each* $a \in \Sigma$ *and process* P

$$a \circ P = P \trianglelefteq a \trianglerighteq P.$$

2.3 Assigning a behavior in BPPA to a program object in PGA

The *behavioral extraction* operator $|-|$ assigns a behavior to a program object.

Definition 4. *For finite program objects the behavior is given by*

$$|X| = |X; (\#0)^\omega|.$$

Definition 5. *The behavior* $|X|$ *of an (infinite) program object* X *is determined recursively by the* **behavior extraction equations** *below:*

$$\begin{aligned}
|!; X| &= S, \\
|a; X| &= a \circ |X|, \\
|+a; u; X| &= |u; X| \trianglelefteq a \trianglerighteq |X|, \\
|-a; u; X| &= |X| \trianglelefteq a \trianglerighteq |u; X|, \\
|\#0; X| &= D, \\
|\#1; X| &= |X|, \\
|\#k+2; u; X| &= |\#k+1; X|.
\end{aligned}$$

By means of these equations, successive steps of the behavior of a program object can be obtained. In the case that a program has a non-trivial loop in which no actions occur, its behavior will be identified with D. Phrased differently: if for a behavior $|X|$ the behavior extraction equations fail to prove $|X| = S$ or $|X| = P \trianglelefteq a \trianglerighteq Q$ for some $a \in \Sigma$ and for some processes P and Q, then $X = D$.

Example 2. The behaviors of program X and Y given in Example 1 are defined as follows.

1. $|X| = c \circ D \trianglelefteq a \trianglerighteq b \circ S$,
2. $|Y| = a \circ b \circ b \circ \cdots$

3 Basic polarized process algebra as cpo's

This section shows that polarized processes can be modeled as a complete partial order (cpo).

To give a denotational semantics for BPPA, we consider a model of BPPA as a solution of the following domain equation

$$\mathcal{P} = \{S, D\} \bigcup (\mathcal{P} \trianglelefteq \Sigma \trianglerighteq \mathcal{P}). \tag{1}$$

where $X \trianglelefteq \Sigma \trianglerighteq Y = \{x \trianglelefteq a \trianglerighteq y | x \in X, y \in Y, a \in \Sigma\}$.

Definition 6. BPPA_Σ *is a set consisting of all finite polarized processes which are made from S and D by means of a finite number of applications of postconditional compositions.*

Intuitively, it can be seen that

Proposition 1. BPPA_Σ *is a model of* BPPA*.*

Proof. Omitted.

Polarized processes can be infinite. To model an infinite process we require a sequence of its finite approximations.

In [6], a technique based on cpo's is described to give a model for BPPA. The main idea of this approach is to define a binary relation $\sqsubseteq$, a partial order, on processes. The expression $P \sqsubseteq Q$ means that P is an approximation of Q. To model all infinite processes the domains of some models of BPPA are required to be *complete*, meaning that every *increasing chain* in these domains has a supremum. It is shown that the set of *projective sequences* is a cpo which contains every finite process in BPPA_Σ. Hence, by completion, every infinite process made by an increasing chain of finite approximations is contained in this model. This implies that the model consisting of projective sequences serves as a semantics for BPPA in a natural way. We review the following definitions and a theorem from [6, 17].

Definition 7.

1. *The* **partial ordering** $\sqsubseteq$ *on* BPPA_Σ *is generated by the clauses*
   (a) for all $P \in \mathrm{BPPA}_\Sigma$, $D \sqsubseteq P$, and
   (b) for all $P, Q, X, Y \in \mathrm{BPPA}_\Sigma$, $a \in \Sigma$,

$$P \sqsubseteq X \,\&\, Q \sqsubseteq Y \Rightarrow P \trianglelefteq a \trianglerighteq Q \sqsubseteq X \trianglelefteq a \trianglerighteq Y.$$

2. *Let $(P_n)_n$ and $(Q_n)_n$ be two sequences in BPPA_Σ, then*

$$(P_n)_n \sqsubseteq (Q_n)_n \Leftrightarrow \forall n \in \mathbb{N} \quad P_n \sqsubseteq Q_n.$$

Definition 8. *An* **increasing chain** $(P_n)_n$ *in* BPPA_Σ *is a sequence satisfying*

$$P_0 \sqsubseteq P_1 \sqsubseteq \cdots \sqsubseteq P_n \sqsubseteq P_{n+1} \sqsubseteq \cdots$$

Definition 9. *A* **complete partial order** *(cpo) $\mathcal{D} = (\mathcal{D}, \sqsubseteq)$ is a partially ordered set with a least element such that every increasing chain has a supremum in $\mathcal{D}$.*

In order to define a projective sequence in BPPA_{Σ}, an operator called *approximation operator* that finitely approximates every process is provided.

Definition 10. *For every $n \in \mathbb{N}$, the* **approximation operator** $\pi_n : \mathrm{BPPA}_{\Sigma} \to \mathrm{BPPA}_{\Sigma}$ *is defined inductively by*

$$\begin{array}{ll} \pi_0(P) & = D, \\ \pi_{n+1}(S) & = S, \\ \pi_{n+1}(D) & = D, \\ \pi_{n+1}(P \trianglelefteq a \trianglerighteq Q) & = \pi_n(P) \trianglelefteq a \trianglerighteq \pi_n(Q), \end{array}$$

A **projective sequence** *is a sequence $(P_n)_{n\in\mathbb{N}}$ such that for each $n \in \mathbb{N}$,*

$$\pi_n(P_{n+1}) = P_n.$$

This definition suggests the existence of a collection of cpo's which are defined as follows

Definition 11. *For all $n \in \mathbb{N}$,* $\mathrm{BPPA}^n_{\Sigma} = \{\pi_n(P) | P \in \mathrm{BPPA}_{\Sigma}\}$

Proposition 2. $\mathrm{BPPA}_{\Sigma} = \bigcup_{n\in\mathbb{N}} \mathrm{BPPA}^n_{\Sigma}$.

Proof. See [6].

Definition 12.

$$\mathrm{BPPA}^{\infty}_{\Sigma} = \{(P_n)_{n\in\mathbb{N}} | P_n \in \mathrm{BPPA}^n_{\Sigma} \,\&\, \pi_n(P_{n+1}) = P_n\}$$

Theorem 1. $(\mathrm{BPPA}^{\infty}_{\Sigma}, \sqsubseteq)$ *is a cpo and* $\mathrm{BPPA}_{\Sigma} \subseteq \mathrm{BPPA}^{\infty}_{\Sigma}$.

Proof. See [6].

4 Basic polarized process algebra as complete ultra-metric spaces

In this section we will illustrate an alternative approach to give a denotational semantics for BPPA. This approach as introduced in [5] is based on the theory of metric spaces. We will define models of BPPA as complete metric spaces and show that a metric space consisting of projective sequences is an appropriate model for BPPA.

We begin by reviewing a few basic concepts of the metric topology from [12].

Definition 13. *A* **metric space** *is a pair (X, d) consisting of a set X and a* **metric** *d on X. The metric $d(x, y)$ defined for arbitrary x and y in X is a nonnegative, real valued function satisfying the conditions:*

1. $d(x,y) = 0$ *if and only if* $x = y$,
2. $d(x,y) = d(y,x)$,
3. $d(x,y) + d(y,z) \geq d(x,z)$.

(X,d) *is said to be an* **ultra-metric space** *if* d *satisfies the strong triangle inequality: For all* $x,y,z \in X$, $d(x,z) \leq \max\{d(x,y), d(y,z)\}$. *We note that for all* $x,y,z \in X$,

$$d(x,z) \leq \max\{d(x,y), d(y,z)\} \Rightarrow d(x,y) + d(y,z) \geq d(x,z).$$

Definition 14. *Metric spaces* (X, d_X) *and* (Y, d_Y) *are said to be* **isometric** *if there is a bijection* $f : X \to Y$ *such that* $d_X(x,y) = d_Y(f(x), f(y))$ *for all* $x,y \in X$.

We shall identify isometric spaces.

Definition 15. $(x_n)_n$ *is a* **Cauchy sequence** *on the space* (X,d) *if*

$$\forall \epsilon > 0 \, \exists N \, \forall n,m > N \quad d(x_n, x_m) < \epsilon.$$

Definition 16. *If every Cauchy sequence in the metric space* R *converges to an element in* R, R *is said to be* **complete**.

Definition 17. *An* ϵ-**neighborhood** *of a point* x *in the metric space* (X,d) *is the set of all points* $y \in X$ *which satisfy the condition* $d(x,y) < \epsilon$. *The point* x *is called a* **contact point** *of* (X,d) *if all its neighborhoods contain at least one point of* (X,d).

Definition 18. *The set of all contact points of a metric space* R *is denoted by* $[R]$ *and is called the* **closure** *of* R.

Definition 19. *Let* R *be an arbitrary metric space. A complete metric space* R^* *is said to be the* **completion** *of the space* R *if:*

1. R *is a subspace of* R^*,
2. R *is everywhere dense in* R^*, *i.e.,* $[R] = R^*$.

It is shown in [12] that the space containing R, together with all limits of its Cauchy sequences is a completion of R, where the distance between the limit points $x^* = \lim_{n\to\infty} x_n$ and $y^* = \lim_{n\to\infty} y_n$ of R is defined as $d(x^*, y^*) = \lim_{n\to\infty} d(x_n, y_n)$. Furthermore, all completions of R are *isometric*.

We will now define a distance between two processes in BPPA_Σ.

Definition 20.

1. $d(S,S) = 0$, $d(D,D) = 0$,
   $d(P,P') = 1$ *if* $P \in \{S,D\}$ *and* $P' \neq P$ *with* $P' \in \mathrm{BPPA}_\Sigma$ *or vice versa,*

2. $d(P_1 \trianglelefteq a_1 \trianglerighteq P_2, Q_1 \trianglelefteq a_2 \trianglerighteq Q_2) =$
$$\begin{cases} 1 & if \quad a_1 \neq a_2, \\ \frac{1}{2}\max\{d(P_1,Q_1), d(P_2,Q_2)\} & otherwise \end{cases}$$
with $P_1, Q_1, P_2, Q_2 \in \mathrm{BPPA}_\Sigma$.

Proposition 3. *For all* $n \in \mathbb{N}$, *(*$\mathrm{BPPA}^n_\Sigma, d$*) is an ultra-metric space.*

Proof. We employ induction on n. The case $n = 0$ is trivial. Assume that ($\mathrm{BPPA}^k_\Sigma, d$) is an ultra-metric space for all $k \leq n$. We will show that ($\mathrm{BPPA}^{n+1}_\Sigma, d$) is also an ultra-metric space. Let $P, Q \in \mathrm{BPPA}^{n+1}_\Sigma$.

1. If $P = Q$ then clearly $d(P,Q) = 0$. Suppose that $d(P,Q) = 0$, we will show that $P = Q$. The case $P \in \{S, D\}$ or $Q \in \{S, D\}$ is trivial. If $P = P_1 \trianglelefteq a_1 \trianglerighteq P_2$ and $Q = Q_1 \trianglelefteq a_2 \trianglerighteq Q_2$ then $P_1, Q_1, P_2, Q_2 \in \mathrm{BPPA}^n_\Sigma$. Since $d(P,Q) = 0$, $a_1 = a_2$. Thus,
$$d(P,Q) = \frac{1}{2}\max\{d(P_1,Q_1), d(P_2,Q_2)\}.$$
This implies that $d(P_1,Q_1) = d(P_2,Q_2) = 0$. Applying the induction hypothesis, we have $P_1 = Q_1$ and $P_2 = Q_2$. Therefore, $P = Q$.
2. $d(P,Q) = d(Q,P)$ follows from the definition.
3. It will be shown that $d(P,Q) \leq \max\{d(P,R), d(R,Q)\}$. If P or Q or R $\in \{S, D\}$, this is trivial. If $P = P_1 \trianglelefteq a_1 \trianglerighteq P_2$, $Q = Q_1 \trianglelefteq a_2 \trianglerighteq Q_2$ and $R = R_1 \trianglelefteq a_3 \trianglerighteq R_2$, then $P_1, P_2, Q_1, Q_2, R_1, R_2 \in \mathrm{BPPA}^n_\Sigma$. There are two cases for a_1, a_2 and a_3:
   (a) $a_3 \neq a_1$ or $a_3 \neq a_2$. Then the right-hand side equals 1. Hence it is always greater or equal than the left-hand side.
   (b) $a_1 = a_2 = a_3$. Then it follows from the induction hypothesis that
   $$\max\{d(P_1,Q_1), d(P_2,Q_2)\} \leq \\ \max\{d(P_1,R_1), d(P_2,R_2), d(R_1,Q_1), d(R_2,Q_2\}.$$
   Thus $d(P,Q) \leq \max\{d(P,R), d(R,Q)\}$.

Therefore, ($\mathrm{BPPA}^n_\Sigma, d$) is an ultra-metric space.

Proposition 4. BPPA_Σ *is an ultra-metric space.*

Proof. This follows from Proposition 2 and Proposition 3.

Since (BPPA_Σ, d) is a metric space, it has a completion, say ($\mathrm{BPPA}^*_\Sigma, d$), which consists of all limits of Cauchy sequences in (BPPA_Σ, d).

Definition 21. *If P and Q are two processes which are represented by the Cauchy sequences $(P_n)_n$ and $(Q_n)_n$ in* BPPA_Σ *then*
$$d(P,Q) = \lim_{n\to\infty} d(P_n, Q_n).$$
P and Q are said to be **equivalent** *if $d(P,Q) = 0$.*

It will be shown that $(\mathrm{BPPA}^*_\Sigma, d)$ is a solution of (1), i.e.,

Lemma 1.

$$\mathrm{BPPA}^*_\Sigma = \{S, D\} \bigcup (\mathrm{BPPA}^*_\Sigma \trianglelefteq \Sigma \trianglerighteq \mathrm{BPPA}^*_\Sigma).$$

Proof.

1. $(\supseteq)$: Since $\{S, D\} \subseteq \mathrm{BPPA}_\Sigma$, $\{S, D\} \subseteq \mathrm{BPPA}^*_\Sigma$. We prove that if $P, Q \in \mathrm{BPPA}^*_\Sigma$ then $(P \trianglelefteq a \trianglerighteq Q) \in \mathrm{BPPA}^*_\Sigma$. Since $P, Q \in \mathrm{BPPA}^*_\Sigma$, $P = \lim_{n\to\infty} P_n$, $Q = \lim_{n\to\infty} Q_n$ for some Cauchy sequences $(P_n)_n$ and $(Q_n)_n$. It is not hard to see that $(P_n \trianglelefteq a \trianglerighteq Q_n)_n$ is also a Cauchy sequence and $P \trianglelefteq a \trianglerighteq Q = \lim_{n\to\infty} P_n \trianglelefteq a \trianglerighteq Q_n$. Thus, $P \trianglelefteq a \trianglerighteq Q \in \mathrm{BPPA}^*_\Sigma$.
2. $(\subseteq)$: If $P \in \mathrm{BPPA}^*_\Sigma$ then $P = S$ or $P = D$ or $P = Q \trianglelefteq a \trianglerighteq R$, $Q, R \in \mathrm{BPPA}^*_\Sigma$. We only consider the case $P \notin \{S, D\}$. Since $P \in \mathrm{BPPA}^*_\Sigma$, $P = \lim_{n\to\infty} P_n$ for some Cauchy sequence $(P_n)_n$. Without lack of generality we can assume that for all n, $P_n = Q_n \trianglelefteq a \trianglerighteq R_n$. Since $(P_n)_n$ is a Cauchy sequence and $d(P_n, P_m) = \frac{1}{2}\max\{d(Q_n, Q_m), d(R_n, R_m)\}$, $(Q_n)_n$ and $(R_n)_n$ are also Cauchy sequences. Therefore, there exist Q and R in BPPA^*_Σ such that $Q = \lim_{n\to\infty} Q_n$, $R = \lim_{n\to\infty} R_n$. Hence $P = Q \trianglelefteq a \trianglerighteq R$.

Lemma 1 indicates that the completion $(\mathrm{BPPA}^*_\Sigma, d)$ of $(\mathrm{BPPA}_\Sigma, d)$ is a model for BPPA. The problem with this model is that each process in $(\mathrm{BPPA}^*_\Sigma, d)$ can be represented by many equivalent Cauchy sequences. We will show that the metric space $(\mathrm{BPPA}^\infty_\Sigma, d)$ achieves a model consisting of all representations from equivalent Cauchy sequences in a unique way. First, we provide some supporting results.

Lemma 2. *For all $P \in \mathrm{BPPA}_\Sigma$ and for all $n \in \mathbb{N}$, $\pi_n(\pi_{n+1}(P)) = \pi_n(P)$.*

Proof. This is proven by induction on n.

Lemma 3. *For all $(P_n)_n \in \mathrm{BPPA}^\infty_\Sigma$, $n \in \mathbb{N}$ and for all $k < n$, $P_k = \pi_k(P_n)$.*

Proof. We employ induction on n.

1. If $n = 0$ then it follows from Definition 12 that $P_0 = \pi_0(P_1)$.
2. $n > 0$. Assume that for all $k < n$, $P_k = \pi_k(P_n)$. It will be shown that for all $k < n + 1$, $P_k = \pi_k(P_{n+1})$. We distinguish three cases:
   (a) If $P_{n+1} = D$ then for all $k < n + 1$, $P_k = \pi_k(D) = D$.
   (b) If $P_{n+1} = S$ then $P_0 = \pi_0(S) = D$ and for all $0 < k < n + 1$, $P_k = \pi_k(S) = S$.
   (c) If $P_{n+1} = P_1 \trianglelefteq a \trianglerighteq P_2$ then it follows from Definition 12 that

   $$P_n = \pi_n(P_{n+1}) = \pi_{n-1}(P_1) \trianglelefteq a \trianglerighteq \pi_{n-1}(P_2).$$

   Therefore, $P_{n-1} = \pi_{n-2}(\pi_{n-1}(P_1)) \trianglelefteq a \trianglerighteq \pi_{n-2}(\pi_{n-1}(P_2))$. Applying Lemma 2, we have $P_{n-1} = \pi_{n-1}(P_{n+1})$ and so on. Thus, for all $k < n+1$, $P_k = \pi_k(P_{n+1})$.

Lemma 4. *For all* $P, Q \in \mathrm{BPPA}_\Sigma$ *and for all* $n \in \mathbb{N}$,

$$d(\pi_n(P), \pi_n(Q)) \le d(\pi_{n+1}(P), \pi_{n+1}(Q))$$

Proof. Omitted.

Proposition 5. *For all* $(P_n)_n, (Q_n)_n \in \mathrm{BPPA}_\Sigma^\infty$, $d(P_n, Q_n)$ *is a non-decreasing sequence. Therefore,*

$$\lim_{n\to\infty} d(P_n, Q_n) = \bigsqcup_{n\in\mathbb{N}} d(P_n, Q_n).$$

Proof. We show that for all $n \in \mathbb{N}$, $d(P_n, Q_n) \le d(P_{n+1}, Q_{n+1})$. For each n, it follows from Lemma 3 that there exist $P, Q \in \mathrm{BPPA}_\Sigma$ such that for all $k \le n+1$, $P_k = \pi_k(P)$, $Q_k = \pi_k(Q)$. By Lemma 4,

$$d(P_n, Q_n) = d(\pi_n(P), \pi_n(Q)) \le d(\pi_{n+1}(P), \pi_{n+1}(Q)) = d(P_{n+1}, Q_{n+1}).$$

Lemma 5. *Let* $P, Q \in \mathrm{BPPA}_\Sigma$. *Then for all* $n \in \mathbb{N}$,

$$d(P, Q) \le \frac{1}{2^n} \Leftrightarrow \pi_n(P) = \pi_n(Q).$$

Proof. This is proven by induction on n.

Proposition 6. *Every element of* $\mathrm{BPPA}_\Sigma^\infty$ *is a Cauchy sequence.*

Proof. Let $(P_n)_n$ be a element in $\mathrm{BPPA}_\Sigma^\infty$. By Lemma 3, for all $m, n \in \mathbb{N}$, $m > n > 0$, $P_{n-1} = \pi_{n-1}(P_n) = \pi_{n-1}(P_m)$. Therefore, by Lemma 5, $d(P_n, P_m) \le \frac{1}{2^{n-1}}$. This implies that $(P_n)_n$ is a Cauchy sequence.

Lemma 6. *Let* Q *be an element in* BPPA_Σ^*. *Then there exists* P *in* $\mathrm{BPPA}_\Sigma^\infty$ *such that* $P = Q$.

Proof. Since Q is an element in BPPA_Σ^*, $Q = \lim_{n\to\infty} Q_n$ for some Cauchy sequence $(Q_n)_n$. To see that there exists $P \in \mathrm{BPPA}_\Sigma^\infty$ such that $P = Q$, we will choose a sequence of natural numbers $N_0, N_1, \ldots$ such that $\pi_n(\pi_{n+1}(Q_{N_{n+1}})) = \pi_n(Q_{N_n})$. Let $P_n = \pi_n(Q_{N_n})$ for all $n \in \mathbb{N}$. Then $P = (P_n)_n$ is an element of $\mathrm{BPPA}_\Sigma^\infty$. We claim that $d(P, Q) = 0$.
Since $(Q_n)_n$ is a Cauchy sequence, we have that

$$\forall \epsilon > 0 \, \exists N \in \mathbb{N} \, \forall m, n > N \quad d(Q_m, Q_n) < \epsilon.$$

– Let $\epsilon = \frac{1}{2^0}$. Then there exists $N_0 \in \mathbb{N}$ such that for all $m, n \ge N_0$, $d(Q_m, Q_n) < \frac{1}{2^0}$. It follows from Lemma 5 that for all $n \ge N_0$,

$$Q_n \in S_0 = \{Q \in \mathrm{BPPA}_\Sigma | \pi_0(Q) = \pi_0(Q_{N_0})\}.$$

– Let $\epsilon = \frac{1}{2^1}$. Then there exists $N_1 \in \mathbb{N}$ such that for all $m, n \geq N_1$, $d(Q_m, Q_n) < \frac{1}{2^1}$. Thus, for all $n \geq N_1$,

$$Q_n \in S_1 = \{Q \in \mathrm{BPPA}_\Sigma | \pi_1(Q) = \pi_1(Q_{N_1})\}.$$

Since Q_{N_1} is also in S_0, $\pi_0(Q_{N_1}) = \pi_0(Q_{N_0})$. By Lemma 2, $\pi_0(\pi_1(Q_{N_1})) = \pi_0(Q_{N_1}) = \pi_0(Q_{N_0})$.

In this way, we can choose a sequence of natural numbers $N_0, N_1, \ldots$ such that $\pi_n(\pi_{n+1}(Q_{N_{n+1}})) = \pi_n(Q_{N_n})$. To see that $d(P, Q) = 0$, consider $m, n \in \mathbb{N}$ such that $m > \max\{N_n, n\}$. Then $\pi_n(Q_m) = \pi_n(Q_{N_n}) = P_n = \pi_n(P_m)$. Thus, $d(P_m, Q_m) < \frac{1}{2^n}$. Hence
$\lim_{m\to\infty} d(P_m, Q_m) = 0$ or $d(P, Q) = 0$.

It follows from Proposition 6 and Lemma 6 that

Theorem 2. $(\mathrm{BPPA}^\infty_\Sigma, d)$ *is the completion of* $(\mathrm{BPPA}_\Sigma, d)$.

In addition, pointwise equal processes in $(\mathrm{BPPA}^\infty_\Sigma, d)$ are identified. That is,

Proposition 7. *For all* $P = (P_n)_n$, $Q = (Q_n)_n$ *in* $(\mathrm{BPPA}^\infty_\Sigma, d)$,

$$P = Q \Leftrightarrow \forall n \in \mathbb{N}\, P_n = Q_n.$$

Proof. This follows from Proposition 5.

The previous results show that $\mathrm{BPPA}^\infty_\Sigma$ is an appropriate model for BPPA. We call this model the **projective limit model** of BPPA.

5 The uniqueness of regular processes in BPPA

Regular processes are investigated in various concurrency theories [14, 13, 8]. In this section, based on Banach's fixed point theorem, we will show that the specification of regular processes as defined in [11] has a unique solution. This suggests the existence of a model consisting of regular processes for BPPA.

Let us recall the definition of *regular processes* from [11] and a few basic concepts of fixed points from [15].

Definition 22. *Let Σ be a set of actions. A process P is* **regular** *over Σ if $P = E_1$, where E_1 is defined by a finite system of the form ($n \geq 1$):*

$$\{E_i = t_i | 1 \leq i \leq n, t_i = S \text{ or } t_i = D \text{ or } t_i = E_{i,r} \trianglelefteq a_i \trianglerighteq E_{i,l}\}$$

with $E_{i,r}, E_{i,l} \in \{E_1, \ldots, E_n\}$ and $a_i \in \Sigma$.

Definition 23. *An element $x \in X$ is said to be a* **fixed point** *of a function $f : X \to X$ if $f(x) = x$.*

Definition 24. *Let (X, d) be a metric space. A function $f : X \to X$ is a* **contraction mapping** *if there is a real number $c < 1$ such that $d(f(x), f(y)) < c \cdot d(x, y)$ for each $x, y \in X$.*

Theorem 3. *(***Banach***) Every contraction mapping of a complete metric space has a unique fixed point.*

Proof. See [15].

We extend the metric on a set M to M^n ($n \geq 1$) as follows

Definition 25. *Let* (M, d) *be a metric space. Let* $X, Y \in M^n$ *for some* $n \geq 1$, $X = [X_1, \ldots, X_n]$, $Y = [Y_1, \ldots, Y_n]$. *Then*

$$d(X, Y) = \max_{i \leq n} d(X_i, Y_i).$$

Then, it is not hard to see that

Proposition 8. *If* (M, d) *is complete then so is* (M^n, d) *for all* $n \geq 1$.

Proof. Omitted.

We will now consider a regular process as a component of the solution of the equation $X = T(X)$, where the definition of T is given as follows

Definition 26. *Let* $T : (\mathrm{BPPA}_\Sigma^\infty)^n \to (\mathrm{BPPA}_\Sigma^\infty)^n$ *be defined such that*

$$T = \lambda X.[t_1(X), \ldots, t_n(X)]$$

where

$$\begin{array}{ll} t_i = \lambda X_1, \ldots, X_n.S & \text{or} \\ t_i = \lambda X_1, \ldots, X_n.D & \text{or} \\ t_i = \lambda X_1, \ldots, X_n.X_{i,l} \trianglelefteq a_i \trianglerighteq X_{i,r} & \end{array}$$

with $X_{i,l}, X_{i,r} \in \{X_1, \ldots, X_n\}$.

Theorem 4. T *has a unique fixed point.*

Proof. Let I be the set of all indexes i such that $t_i = X_{i,l} \trianglelefteq a_i \trianglerighteq X_{i,r}$. To show uniqueness, let X, Y be elements of $(\mathrm{BPPA}_\Sigma^\infty)^n$. We note that $d(t_i(X), t_i(Y)) = 0$ if $i \notin I$, since $t_i(X)$ is a constant, and
$d(t_i(X), t_i(Y)) = \frac{1}{2} \max\{d(X_{i,l}, Y_{i,l}), d(X_{i,r}, Y_{i,r})\}$ otherwise. Then by Definition 25 we have

$$\begin{aligned} d(T(X), T(Y)) &= \max_{i \leq n} d(t_i(X), t_i(Y)) \\ &= \max_{i \in I} (\tfrac{1}{2} \max\{d(X_{i,l}, Y_{i,l}), d(X_{i,r}, Y_{i,r})\}) \\ &\leq \tfrac{1}{2} \max_{i \leq n} d(X_i, Y_i) = \tfrac{1}{2} d(X, Y). \end{aligned}$$

From Definition 24 it follows that T is a contraction mapping. Since $\mathrm{BPPA}_\Sigma^\infty$ is complete, it follows from Proposition 8 that $(\mathrm{BPPA}_\Sigma^\infty)^n$ is also complete. By Banach's fixed point theorem T has a unique solution.

The previous theorem implies that the specification of a regular process determines a unique process. This suggests the following definition.

Definition 27. $\mathrm{BPPA}_r/_=$ *is the set of regular processes in* $\mathrm{BPPA}_\Sigma^\infty$ *modulo equivalence.*

Proposition 9. $\mathrm{BPPA}_r/_=$ *is a model of* BPPA.

Proof. Omitted.

6 Comparing the models of BPPA

In this section we compare several models of BPPA. First of all, the two models $(\mathrm{BPPA}^{\infty}_{\Sigma}, \sqsubseteq)$ and $(\mathrm{BPPA}^{\infty}_{\Sigma}, d)$ are compared by means of *compatibility*. After that, we introduce another model, denoted by $\mathrm{BPPA}^{\omega}_{\Sigma}$, consisting of all Cauchy sequences in BPPA_{Σ}. The extensions of the metric spaces $(\mathrm{BPPA}^{\infty}_{\Sigma}, d)$ and ($\mathrm{BPPA}^{\omega}_{\Sigma}$,d) with the *abstraction* operator (see [2]) are discussed.

We use the following definition from [4].

Definition 28. *A cpo* $(\mathbb{D}, \sqsubseteq)$ *and a complete metric space* $(\mathbb{M}, d)$ *are said to be* **compatible** *if* $\mathbb{D} = \mathbb{M}$ *and* $\bigsqcup_n x_n = \lim_{n\to\infty} x_n$ *for each monotone Cauchy sequence* $(x_n)_n$.

Proposition 10. *Let* $(P_n)_n$ *be an increasing chain in* BPPA_{Σ}. *Then*

$$\forall n\, \exists N\, \forall m > N \quad \pi_n(P_m) = \pi_n(P_N).$$

Proof. We distinguish two cases. If for all m, $P_m \in \{D, S\}$ then there exists a minimal N such that for all $m > N$, $P_m = P_N$. Thus, for all n, $\pi_n(P_m) = \pi_n(P_N)$. The other case is that there exists a minimal N_0 such that for all $m \geq N_0$, $P_m = Q_m \trianglelefteq a \trianglerighteq R_m$. It is not hard to see that $(Q_m)_m$ and $(R_m)_m$ are also increasing chains. We note that for all $m < N_0$, $Q_m = R_m = D$. We employ induction on n.

1. If $n = 0$ then clearly $N = 1$.
2. If $n > 0$ then for all $m \geq N_0$, $\pi_n(P_m) = \pi_{n-1}(Q_m) \trianglelefteq a \trianglerighteq \pi_{n-1}(R_m)$. Applying the induction hypothesis, there exist N_1 and N_2 such that for all $m > N_1$, $\pi_{n-1}(Q_m) = \pi_{n-1}(Q_{N_1})$ and for all $m > N_2$, $\pi_{n-1}(R_m) = \pi_{n-1}(R_{N_2})$. Let $N = \max\{N_0, N_1, N_2\}$. Then for all $m > N$, $\pi_n(P_m) = \pi_n(P_N)$.

Therefore, for all $n \in \mathbb{N}$, there exists $N \in \mathbb{N}$ such that for all $m > N$, $\pi_n(P_m) = \pi_n(P_N)$.

Lemma 7. *Every increasing chain* $(P_n)_n$ *in* BPPA_{Σ} *is a Cauchy sequence and* $\bigsqcup_n P_n = \lim_{n\to\infty} P_n$.

Proof. Let $P = \bigsqcup_n P_n$. It follows from Proposition 10 and Lemma 5 that for all $n \in \mathbb{N}$, there exists $N \in \mathbb{N}$ such that for all $m > N$, $d(P_m, P) < \frac{1}{2^n}$. This implies that $\lim_{n\to\infty} P_n = P$. Therefore, $(P_n)_n$ is a Cauchy sequence and $\bigsqcup_n P_n = \lim_{n\to\infty} P_n$.

It follows that

Lemma 8. *(*$\mathrm{BPPA}^{\infty}_{\Sigma}, \sqsubseteq$*) and (*$\mathrm{BPPA}^{\infty}_{\Sigma}, d$*) are compatible.*

We now compare the two complete metric spaces $(\mathrm{BPPA}^{\infty}_{\Sigma}, d)$ and $(\mathrm{BPPA}^{\omega}_{\Sigma}, d)$ by investigating the continuity property of the abstraction operator in these spaces.

Definition 29.

$$\mathrm{BPPA}_{\Sigma}^{\omega} = \{(P_n)_{n\in\mathbb{N}} | (P_n)_n \textit{is a Cauchy sequence in } \mathrm{BPPA}_{\Sigma}\}$$

By Proposition 6 and Lemma 6, we have that

Proposition 11. $P \in (\mathrm{BPPA}_{\Sigma}^{\infty}, d) \Leftrightarrow P \in (\mathrm{BPPA}_{\Sigma}^{\omega}, d)$.

Assume that there exists a basic internal action $t \in \Sigma$ which does not have any side effects and always replies *true*. This action can be abstracted by an operator called *abstraction operator* which replaces occurrences of t by silent steps. We recall the following definitions and a lemma of abstraction from [7].

Definition 30.

1. *Let* $\tau \circ \_ : \mathrm{BPPA}_{\Sigma} \to \mathrm{BPPA}_{\Sigma}$ *be defined by*

$$\tau \circ P = P.$$

2. *Let* $\tau_t : \mathrm{BPPA}_{\Sigma} \to \mathrm{BPPA}_{\Sigma}$ *be defined by*

$$\begin{array}{ll} \tau_t(S) & = S, \\ \tau_t(D) & = D, \\ \tau_t(P \trianglelefteq t \trianglerighteq Q) & = \tau \circ \tau_t(P), \\ \tau_t(P \trianglelefteq a \trianglerighteq Q) & = \tau_t(P) \trianglelefteq a \trianglerighteq \tau_t(Q) \quad (a \neq t \in \Sigma). \end{array}$$

In [7], the authors show that the abstraction operator is monotone, i.e.,

Lemma 9. *For all* $P, Q \in \mathrm{BPPA}_{\Sigma}$,

$$P \sqsubseteq Q \Leftrightarrow \tau_t(P) \sqsubseteq \tau_t(Q).$$

Proof. See [7].

Definition 31. *For* $P = (P_n)_{n\in\mathbb{N}} \in \mathrm{BPPA}_{\Sigma}^{\infty}$, *let*

$$\tau_t(P) = \bigsqcup_n \tau_t(P_n).$$

It will be shown that abstraction can be easily extended to the metric space $(\mathrm{BPPA}_{\Sigma}^{\infty}, d)$, i.e.,

Proposition 12. *For all elements* P *of the metric space* $\mathrm{BPPA}_{\Sigma}^{\infty}$,

$$\tau_t(P) \in (\mathrm{BPPA}_{\Sigma}^{\infty}, d).$$

Proof. Let $P = (P_n)_n$. Then for all n,

$$P_n = \pi_n(P_{n+1}) \sqsubseteq P_{n+1}.$$

By Lemma 9, $(\tau_t(P_n))_n$ is an increasing chain. It follows from Lemma 7 that $\tau_t(P) \in (\mathrm{BPPA}^{\omega}_{\Sigma}, d)$. By Proposition 11, $\tau_t(P) \in (\mathrm{BPPA}^{\infty}_{\Sigma}, d)$.

This means that abstraction is continuous in $(\mathrm{BPPA}^{\infty}_{\Sigma}, d)$. We will now consider the continuity property of this operator in $(\mathrm{BPPA}^{\omega}_{\Sigma}, d)$.

Proposition 13. *The abstraction operator is not continuous in* $(\mathrm{BPPA}^{\omega}_{\Sigma}, d)$*, i.e., there exists an element* $P = (P_n)_n$ *of* $(\mathrm{BPPA}^{\omega}_{\Sigma}, d)$ *such that* $\lim_{n\to\infty} \tau_t(P_n) \neq \tau_t(P)$.

Proof. Let $(P_n)_n$ be defined as follows

$$(P_n)_n = D, t \circ S, t^2 \circ D, \ldots, t^{2n} \circ D, t^{2n+1} \circ S, \ldots$$

It is not hard to see that $(P_n)_n$ is a Cauchy sequence. Let $P = (P_n)_n$. Then $P \in (\mathrm{BPPA}^{\omega}_{\Sigma}, d)$. Therefore, by Proposition 11 and Proposition 12, there exists $\tau_t(P)$ in the space $(\mathrm{BPPA}^{\omega}_{\Sigma}, d)$. However, the sequence

$$(\tau_t(P_n))_n = D, S, D, \ldots, D, S, \ldots$$

is not a Cauchy sequence. Thus, it does not have a limit in $(\mathrm{BPPA}^{\omega}_{\Sigma}, d)$. Therefore, $\lim_{n\to\infty} \tau_t(P_n) \neq \tau_t(P)$.

7 Concluding remarks

In this paper, we have considered various issues in giving a denotational semantics for BPPA, a formal semantics for PGA.

We have presented several models for BPPA. Based on the methodology of [5], we have shown that the projective limit model $\mathrm{BPPA}^{\infty}_{\Sigma}$ is an appropriate model for BPPA. The advantage of $\mathrm{BPPA}^{\infty}_{\Sigma}$ is that it comprises also infinite processes in a unique way and can be easily extended with the abstraction operator. Furthermore, we have proved that the specification of a regular process has a unique solution. This indicates the existence of the model $\mathrm{BPPA}_r/_=$ consisting of regular processes modulo equivalence in $\mathrm{BPPA}^{\infty}_{\Sigma}$. To compare the various models of BPPA, it has been proved that $(\mathrm{BPPA}^{\infty}_{\Sigma}$,d) and the model based on complete partial orders $(\mathrm{BPPA}^{\infty}_{\Sigma}, \sqsubseteq)$ are compatible. We also have shown that the model consisting of Cauchy sequences $\mathrm{BPPA}^{\omega}_{\Sigma}$ cannot be extended with the abstraction operator in a natural way.

Finally, it should be noted that the model BPPA_{Σ} is a submodel of $\mathrm{BPPA}_r/_=$ while $\mathrm{BPPA}_r/_=$, in turn, is a submodel of $\mathrm{BPPA}^{\infty}_{\Sigma}$.

Acknowledgment

I thank Inge Bethke for her stimulating discussions and constructive comments.

References

1. P. America and J. Rutten. Solving recursive domain equations in a category of complete metric spaces. *J. Computer and System Sciences*, 39 (3):343–375, 1989.
2. J.C.M. Baeten and W.P. Weijland. *Process Algebra*. Cambridge Tracts in Theoretical Computer Science 18. Cambridge University Press, 1990.
3. C. Baier and M. Majster-Cederbaum. Denotational semantics in the cpo and metric approach. *Theoretical Computer Science*, 135:171–220, 1994.
4. C. Baier and M. Majster-Cederbaum. The connection between initial and unique solutions of domain equations in the partial order and metric approach. *Formal Aspects of Computing*, 9:425–445, 1997.
5. J.W. Bakker and J.I. Zucker. Processes and the denotational semantics of concurrency. *Information and Control*, 54(1/2):70–120, 1982.
6. J.A. Bergstra and I. Bethke. Polarized process algebra and program equivalence. *ICALP*, 2003.
7. J.A. Bergstra and I. Bethke. Polarized process algebra with reactive composition. To appear.
8. J.A. Bergstra and J.W. Klop. The algebra of recursively defined processes and the algebra of regular processes. *ICALP*, 11:82–95, 1984.
9. J.A. Bergstra and M.E. Loots. Program algebra for component code. *Formal Aspects of Computing*, 12:1–17, 2000.
10. J.A. Bergstra and M.E. Loots. Program algebra for sequential code. *J. Logic Algebr. Programming*, 51:125–156, 2002.
11. I. Bethke and A. Ponse. *Programma-algebra, een inleiding tot de programmatuur (in Dutch)*. Amsterdam University Press, Vossiuspers, 2003.
12. A.N. Kolmogorov and S.V. Fomin. *Elements of the theory of functions and functional analysis*. Graylock press, Rochester, N.Y, 1957.
13. G.J. Milne. Abstraction and nondeterminism in concurrent systems. *IEEE*, 3:358–364, 1982.
14. R. Milner. *A calculus of communicating system*. LNCS 92. Springer Verlag, Berlin, 1980.
15. G.E. Shilov. *Elementary Functional Analysis II*. The MIT Press, Cambridge, Massachusetts and London, 1974.
16. M.B. Smyth and G.D. Plotkin. The category-theoretic solution of recursive equation. *SIAM J. Comput.*, 11:761–783, 1982.
17. V. Stoltenberg-Hansen, I. Lindstrom, and E.R. Griffor. *Mathematical Theory of Domains*. Cambridge Tracts in Theoretical Computer Science 22. Cambridge University Press, 1994.
18. J.E. Stoy. *Denotational Semantics: The Scott-Strachey Approach to Programming Language Semantics*. Cambridge University Press, Cambridge, Massachusetts, 1977.

Electronic Reports Series of the Programming Research Group

Within this series the following reports appeared.

[PRG0503] T.D. Vu, *Metric Denotational Semantics for BPPA*, Programming Research Group - University of Amsterdam, 2005.

[PRG0502] J.A. Bergstra, I. Bethke, and A. Ponse, *Decision Problems for Pushdown Threads*, Programming Research Group - University of Amsterdam, 2005.

[PRG0501] J.A. Bergstra and A. Ponse, *A Bypass of Cohen's Impossibility Result*, Programming Research Group - University of Amsterdam, 2005.

[PRG0405] J.A. Bergstra and I. Bethke, *An Upper Bound for the Equational Specification of Finite State Services*, Programming Research Group - University of Amsterdam, 2004.

[PRG0404] J.A. Bergstra and C.A. Middelburg, *Thread Algebra for Strategic Interleaving*, Programming Research Group - University of Amsterdam, 2004.

[PRG0403] B. Diertens, *A Compiler-projection from PGLEc.MSPio to Parrot*, Programming Research Group - University of Amsterdam, 2004.

[PRG0402] J.A. Bergstra and I. Bethke, *Linear Projective Program Syntax*, Programming Research Group - University of Amsterdam, 2004.

[PRG0401] B. Diertens, *Molecular Scripting Primitives*, Programming Research Group - University of Amsterdam, 2004.

[PRG0302] B. Diertens, *A Toolset for PGA*, Programming Research Group - University of Amsterdam, 2003.

[PRG0301] J.A. Bergstra and P. Walters, *Projection Semantics for Multi-File Programs*, Programming Research Group - University of Amsterdam, 2003.

[PRG0201] I. Bethke and P. Walters, *Molecule-oriented Java Programs for Cyclic Sequences*, Programming Research Group - University of Amsterdam, 2002.

The above reports and more are available through the website: www.science.uva.nl/research/prog/

Electronic Report Series

Programming Research Group
Faculty of Science
University of Amsterdam

Kruislaan 403
1098 SJ Amsterdam
the Netherlands

www.science.uva.nl/research/prog/