University of Amsterdam
Programming Research Group

Timed Tuplix Calculus and the Wesseling and van den Berg Equation

J.A. Bergstra
C.A. Middelburg

Report PRG0901 January 2009

J.A. Bergstra

Programming Research Group
Faculty of Science
University of Amsterdam

Kruislaan 403
1098 SJ Amsterdam
The Netherlands

tel. +31 20 525.7591
e-mail: janb@science.uva.nl

C.A. Middelburg

Programming Research Group
Faculty of Science
University of Amsterdam

Kruislaan 403
1098 SJ Amsterdam
The Netherlands

e-mail: kmiddelb@science.uva.nl

Programming Research Group Electronic Report Series

Timed Tuplix Calculus and the Wesseling and van den Bergh Equation*

J.A. Bergstra and C.A. Middelburg

Programming Research Group, University of Amsterdam,
P.O. Box 41882, 1009 DB Amsterdam, the Netherlands
J.A.Bergstra@uva.nl,C.A.Middelburg@uva.nl

Abstract. We formalize a cumulative interest compliant conservation requirement for pure financial products proposed by Wesseling and van den Bergh to make financial issues relating to these products amenable to mathematical analysis. The formalization is given in a timed extension of tuplix calculus and abstracts from the idiosyncrasies of time measurement. We also use the timed tuplix calculus to show how wanted financial behaviours may profit from using pure financial products.

Keywords: timed tuplix calculus, realistic interest calculation axiom, Wesseling and van den Bergh equation, implicit capital, signed cancellation meadow.

1 Introduction

In [10], Wesseling and van den Bergh claim that interest calculations relating to financial products should always be based on cumulative interests. By strictly adhering to the use of cumulative interests, the design of financial products is made symmetric between client and provider and an implicit bias towards either party can be avoided. This is the point of departure of their 'realistic interest calculation approach'. Applying this approach involves a strict separation between transfers related to a financial product proper and transfers related to its costs of delivery. Transfers related to the financial product proper include transfers due to interests. Transfers related to the costs of delivery may include clear profit, general running cost, cost of insurance against non-payment, costs of marketing and communication, etc. Having made this separation, Wesseling and van den Bergh formulate a cumulative interest compliant conservation requirement for pure financial products: the sum of all transfers relating to the product, transposed to some point of time (the focal date) by means of cumulative interest at the actual interest rate of the product, is zero.

Our objective is to formalize this conservation requirement in the setting of tuplix calculus [6], a calculus that has been applied earlier in modular financial budget design. In order to achieve that objective, we introduce a timed

* This research was carried out in the framework of the Jacquard-project Symbiosis, which is funded by the Netherlands Organisation for Scientific Research (NWO).

extension of tuplix calculus. A single equational axiom of the extended calculus expresses that interest calculations are based on cumulative interests. We formalize the above-mentioned conservation requirement for pure financial products by an equation in the extended calculus. Unaware of previous occurrences of the requirement in the financial literature, we call this equation the Wesseling and van den Bergh equation. Moreover, we adapt the notion of implicit capital of a process introduced in [4] to the setting of timed tuplices and use it to show how wanted financial behaviours may profit from using pure financial products. The implicit capital associated with a financial behaviour may be viewed as the least amount of money that must be at disposal to exhibit that behaviour – taking cumulative interest into account.

In [6], tuplix calculus is presented by first introducing a core tuplix calculus and after that extending the core tuplix calculus with several operators. In the timed tuplix calculus introduced in this paper, only the core tuplix calculus is included. In the core tuplix calculus as well as its extensions, the mathematical structure for quantities is a signed cancellation meadow [5]. The prime example of cancellation meadows is the field of rational numbers with the multiplicative inverse operation made total by imposing that the multiplicative inverse of zero is zero. A cancellation meadow is an appropriate mathematical structure for quantities if quantities are measured with finite accuracy. A signed cancellation meadow is a cancellation meadow expanded with a signum operation.

This paper is organized as follows. First, we give a brief summary of signed cancellation meadows (Section 2). Next, we review the core of tuplix calculus (Section 3). Then, we extend the core of tuplix calculus to a timed tuplix calculus (Section 4). After that, we formalize the conservation requirement for pure financial products and show how financial behaviours may profit from using pure financial products (Section 5). Following this, we construct the standard model of the timed tuplix calculus (Section 6). Finally, we make some concluding remarks (Section 7).

2 Signed Cancellation Meadows

In the timed tuplix calculus presented in this paper, the mathematical structure for quantities is a signed cancellation meadow. Signed cancellation meadows were first introduced in [5]. In this section, we give a brief summary of signed cancellation meadows.

A meadow is a commutative ring with identity equipped with a multiplicative inverse operation satisfying a reflexivity equation and a restricted inverse equation, and in which the multiplicative inverse of zero is zero. A cancellation meadow is a meadow in which the multiplicative inverse operation satisfies a general inverse law. A signed meadow is a meadow expanded with a signum operation. Meadows are defined for the first time in [7]. In that paper, cancellation meadows are called zero-totalized field. The expansion of meadows with a signum operation originates from [5].

The signature of meadows consists of the following constants and operators:

Table 1. Equations for meadows

$(u + v) + w = u + (v + w)$	$(u \cdot v) \cdot w = u \cdot (v \cdot w)$	$(u^{-1})^{-1} = u$
$u + v = v + u$	$u \cdot v = v \cdot u$	$u \cdot (u \cdot u^{-1}) = u$
$u + 0 = u$	$u \cdot 1 = u$	
$u + (-u) = 0$	$u \cdot (v + w) = u \cdot v + u \cdot w$	

- the constants 0 and 1;
- the binary *addition* operator $+$;
- the binary *multiplication* operator $\cdot$;
- the unary *additive inverse* operator $-$;
- the unary *multiplicative inverse* operator $^{-1}$.

We assume that there are infinitely many variables, including u, v and w. Terms are build as usual. We use infix notation for the binary operators $+$ and $\cdot$, prefix notation for the unary operator $-$, and postfix notation for the unary operator $^{-1}$. We use the usual precedence convention to reduce the need for parentheses. We introduce subtraction and division as abbreviations: $p - q$ abbreviates $p + (-q)$ and p/q abbreviates $p \cdot q^{-1}$. We use numerals in the common way (2 abbreviates $1 + 1$, etc.). We also use the notation p^n for exponentiation with a natural number as exponent. For each term p over the signature of meadows, the term p^n is defined by induction on n as follows: $p^0 = 1$ and $p^{n+1} = p^n \cdot p$.

The constants and operators from the signature of meadows are adopted from rational arithmetic, which gives an appropriate intuition about these constants and operators.

A meadow is an algebra over the signature of meadows that satisfies the equations given in Table 1. Thus, a meadow is a commutative ring with identity equipped with a multiplicative inverse operation $^{-1}$ satisfying the reflexivity equation $(u^{-1})^{-1} = u$ and the restricted inverse equation $u \cdot (u \cdot u^{-1}) = u$. From the equations given in Table 1, the equation $0^{-1} = 0$ is derivable.

In meadows, the multiplicative inverse operation is total. The advantage of working with a total multiplicative inverse operation lies in the fact that conditions like $u \neq 0$ in $u \neq 0 \Rightarrow u \cdot u^{-1} = 1$ are not needed to guarantee meaning.

A *non-trivial meadow* is a meadow that satisfies the *separation axiom*

$$0 \neq 1 \ .$$

A *cancellation meadow* is a meadow that satisfies the *cancellation axiom*

$$u \neq 0 \wedge u \cdot v = u \cdot w \Rightarrow v = w \ ,$$

or equivalently, the *general inverse law*

$$u \neq 0 \Rightarrow u \cdot u^{-1} = 1 \ .$$

Table 2. Equations for signum operation

$\mathsf{s}(u/u) = u/u$	$\mathsf{s}(u^{-1}) = \mathsf{s}(u)$
$\mathsf{s}(1 - u/u) = 1 - u/u$	$\mathsf{s}(u \cdot v) = \mathsf{s}(u) \cdot \mathsf{s}(v)$
$\mathsf{s}(-1) = -1$	$(1 - \frac{\mathsf{s}(u)-\mathsf{s}(v)}{\mathsf{s}(u)-\mathsf{s}(v)}) \cdot (\mathsf{s}(u+v) - \mathsf{s}(u)) = 0$

An important property of non-trivial cancellation meadows is the following: $0/0 = 0$, whereas $u/u = 1$ for $u \neq 0$.

A *signed meadow* is a meadow expanded with a unary *signum* operation s satisfying the equations given in Table 2. In combination with the cancellation axiom, the last equation in this table is equivalent to the conditional equation $\mathsf{s}(u) = \mathsf{s}(v) \Rightarrow \mathsf{s}(u+v) = \mathsf{s}(u)$.

In signed cancellation meadows, the function max has a simple definition:

$$\max(u, v) = \frac{\mathsf{s}(u-v)+1}{2} \cdot (u - v) + v \ .$$

Henceforth, we will write $p \leq q$ for $\max(p, q) = q$.

3 Core Tuplix Calculus and Encapsulation

The timed tuplix calculus presented in this paper extends CTC (Core Tuplix Calculus). CTC has been introduced in [6] as the core of TC (Tuplix Calculus). In this section, we give a brief summary of CTC and its extension with encapsulation operators. These operators have been introduced in [6] as well. The operators of the timed tuplix calculus that will be introduced in Section 4 include generalizations of the encapsulation operators.

It is assumed that a fixed but arbitrary set A of *transfer actions* has been given. It is also assumed that a fixed but arbitrary signed non-trivial cancellation meadow $\mathcal{D}$ has been given.

CTC has two sort: the sort $\mathbf{T}$ of *tuplices* and the sort $\mathbf{Q}$ of *quantities*. To build terms of sort $\mathbf{T}$, it has the following constants and operators:

- the *empty tuplix* constant $\epsilon : \mathbf{T}$;
- the *blocking tuplix* constant $\delta : \mathbf{T}$;
- for each $a \in A$, the unary *transfer action* operator $a : \mathbf{Q} \to \mathbf{T}$;
- the unary *zero test* operator $\gamma : \mathbf{Q} \to \mathbf{T}$;
- the binary *conjunctive composition* operator $\mathbin{⦶} : \mathbf{T} \times \mathbf{T} \to \mathbf{T}$.

To build terms of sort $\mathbf{Q}$, CTC has the constants and operators from the signature of meadows.

We assume that there are infinitely many variables of sort $\mathbf{T}$, including x, y and z, and infinitely many variables of sort $\mathbf{Q}$, including u, v and w. Terms are build as usual for a many-sorted signature (see e.g. [9, 11]). We use infix notation for the binary operator $\mathbin{⦶}$.

Table 3. Axioms of CTC

$x \mathbin{⨀} y = y \mathbin{⨀} x$	T1	$\gamma(u) = \gamma(u/u)$	T6
$(x \mathbin{⨀} y) \mathbin{⨀} z = x \mathbin{⨀} (y \mathbin{⨀} z)$	T2	$\gamma(0) = \epsilon$	T7
$x \mathbin{⨀} \epsilon = x$	T3	$\gamma(1) = \delta$	T8
$x \mathbin{⨀} \delta = \delta$	T4	$\gamma(u) \mathbin{⨀} \gamma(v) = \gamma(u/u + v/v)$	T9
$a(u) \mathbin{⨀} a(v) = a(u + v)$	T5	$\gamma(u - v) \mathbin{⨀} a(u) = \gamma(u - v) \mathbin{⨀} a(v)$	T10

A term of sort $\mathbf{T}$ is *tuplix-closed* if it does not contain variables of sort $\mathbf{T}$. A term of sort $\mathbf{T}$ is *closed* if it does not contain variables of any sort.

We look at CTC as a calculus that is concerned with transfers of quantities of something. Let t and t' be closed terms of sort $\mathbf{T}$, and let q be a closed term of sort $\mathbf{Q}$. Intuitively, the constants and operators introduced above can be explained as follows:

- ϵ is a tuplix with no effect;
- δ blocks any joint effect of tuplices;
- the effect of $a(q)$ is performing action a and transferring quantity q on performing that action;
- $\gamma(q)$ is with no effect if q equals 0 and blocks any joint effect otherwise;
- the effect of $t \mathbin{⨀} t'$ is the joint effect of t and t'.

In [6], these constants and operators are explained in a different way. We consider that way of explanation less appropriate for the timed extension of CTC that will be presented in Section 4.

We use the following convention: a transfer of a positive quantity is taken as an outgoing transfer and a transfer of a negative quantity is taken as an incoming transfer.

Notice that CTC can be looked upon as a special purpose process algebra in which processes are considered at a level of detail where not even the order in which actions are performed matter. This makes CTC suitable for formalizing budgets: budgets are in fact descriptions of financial behaviour at the level of detail where only the actions to be performed and the quantities transferred on performing those actions matter.

The axioms of CTC are given in Table 3. The following proof rule is adopted to lift the valid equations between terms of sort $\mathbf{Q}$ to CTC:

$$\text{for all terms } p \text{ and } q \text{ of sort } \mathbf{Q}, \quad \mathcal{D} \models p = q \quad \text{implies} \quad \gamma(p) = \gamma(q) \ .$$

We will refer to this proof rule by DE.

To prove a statement for all CTC terms of sort $\mathbf{T}$, it is is sufficient to prove it for all CTC canonical terms. A *CTC canonical term* is a CTC term of sort $\mathbf{T}$ of the form

$$\gamma(p_0) \mathbin{⨀} a_1(p_1) \mathbin{⨀} \ldots \mathbin{⨀} a_k(p_k) \mathbin{⨀} x_1 \mathbin{⨀} \ldots \mathbin{⨀} x_l \ ,$$

where $k, l \geq 0$ and $a_1, \ldots, a_k$ are distinct transfer actions.

Table 4. Axioms for encapsulation

$\partial_H(\epsilon) = \epsilon$	E1	$\partial_H(a(u)) = a(u)$	if $a \notin H$	E4
$\partial_H(\delta) = \delta$	E2	$\partial_H(a(u)) = \gamma(u)$	if $a \in H$	E5
$\partial_H(\gamma(u)) = \gamma(u)$	E3	$\partial_H(x \trianglerighteq \partial_H(y)) = \partial_H(x) \trianglerighteq \partial_H(y)$		E6
		$\partial_{H \cup H'}(x) = \partial_H(\partial_{H'}(x))$		E7

Proposition 1. *For all* CTC *terms* t *of sort* $\mathbf{T}$*, there exists a* CTC *canonical term* t' *such that* $t = t'$ *is derivable from the axioms of* CTC*.*

Proof. This proposition is a reformulation of Lemma 1 from [6]. □

Like in [6], we can add the following operators to the operators of CTC to build terms of sort $\mathbf{T}$:

– for each $H \subseteq A$, the unary *encapsulation* operator $\partial_H : \mathbf{T} \to \mathbf{T}$.

Let t be a closed term of sort $\mathbf{T}$. Intuitively, the encapsulation operators can be explained as follows:

– if, for each $a \in H$, the sum of all quantities transferred by t on performing a equals 0, then $\partial_H(t)$ differs from t in that, for each $a \in H$, the effect of all transfer actions of the form $a(p)$ occurring in t is eliminated; otherwise, $\partial_H(t)$ has the same effect as δ.

The name encapsulation was introduced earlier in the setting of the process algebra ACP for similar operations in [3].

The axioms for encapsulation are given in Table 4.

4 Timed Tuplix Calculus

In this section, we extend CTC to TTC (Timed Tuplix Calculus). In the informal explanation of the constants and operators of CTC given in Section 3, we could disregard what it is of which quantities are transferred. Clearly, if CTC is used to formalize budgets, quantities of money are transferred. It happens to be far from obvious to give informal explanations of two of the additional operators of TTC that are not couched in terms of quantities of money, usually called amounts of money. Therefore, we change over in this section to explanations couched in terms of amounts of money. This should not be taken as a suggestion that more abstract explanations are impossible. In subsequent sections, tuplices are viewed as representations of financial behaviours. The change-over made in this section agrees with this viewpoint.

Like CTC, TTC has two sort: the sort $\mathbf{T}$ of tuplices and the sort $\mathbf{Q}$ of quantities. To build terms of sort $\mathbf{T}$, it has the constants and operators of CTC to build terms of sort $\mathbf{T}$, and in addition the following operators:

– the unary *delay* operator $\sigma : \mathbf{T} \to \mathbf{T}$;

– for each $I \subseteq A$, the unary *pre-abstraction* operator $\mathtt{t}_I : \mathbf{T} \to \mathbf{T}$;
– for each $H \subseteq A$, the binary *interest counting encapsulation* operator $\partial_H : \mathbf{Q} \times \mathbf{T} \to \mathbf{T}$.

To build terms of sort $\mathbf{Q}$, it has the constants and operators from the signature of meadows, and in addition the following operator:

– the binary *implicit capital* operator $\mathrm{Q} : \mathbf{Q} \times \mathbf{T} \to \mathbf{Q}$.

We write $\partial_H^p(t)$ and $\mathrm{Q}^p(t)$, where p is a term of sort $\mathbf{Q}$ and t is a term of sort $\mathbf{T}$, for $\partial_H(p,t)$ and $\mathrm{Q}(p,t)$, respectively. We also use the notation $\sigma^n(t)$. For each term t of sort $\mathbf{T}$, the term $\sigma^n(t)$ is defined by induction on n as follows: $\sigma^0(t) = t$ and $\sigma^{n+1}(t) = \sigma(\sigma^n(t))$.

In TTC, it is assumed that $\mathtt{t} \in A$. A special role is assigned to $\mathtt{t}$: transfer actions of the form $a(p)$ are renamed to $\mathtt{t}(p)$ on pre-abstraction in order to abstract from their identity, but not from their presence.

Let t be a closed term of sort $\mathbf{T}$ and let p be a closed term of sort $\mathbf{Q}$. Intuitively, the additional operators introduced above can be explained as follows:

– $\sigma(t)$ differs from t in that the effect of each transfer action occurring in t is delayed one time slice;
– $\mathtt{t}_I(t)$ differs from t in that, for each $a \in I$, the effect of each transfer action of the form $a(p)$ occurring in t is replaced by the effect of $\mathtt{t}(p)$;
– $\partial_H^p(t)$ differs from $\partial_H(t)$ in that, for each $a \in H$, a cumulative interest at the rate of p per time slice is taken into account on the summation of all amounts of money transferred by t on performing a;
– $\mathrm{Q}^p(t)$ is the least amount of money that must be at disposal to exhibit financial behaviour t if a cumulative interest at the rate of p per time slice is taken into account.

The delay operator introduced here is comparable to the relative discrete time unit delay operator and the absolute discrete time unit delay operator introduced earlier in the setting of the process algebra ACP in [2]. The pre-abstraction operators introduced here are comparable to the pre-abstraction operators introduced earlier in the setting of the process algebra ACP in [1]. The interest counting encapsulation operators are generalizations of the encapsulation operators introduced in Section 3: $\partial_H(t)$ can be taken as abbreviation of $\partial_H^0(t)$. The implicit capital operator introduced here is comparable to the implicit computational capital operator introduced earlier in the setting of the process algebra ACP in [4].

The implicit capital of a financial behaviour is an amount of money that cannot be negative. However, it is undefined if the behaviour is δ. In order to circumvent the use of algebras with partial operations, -1 is used to represent the undefinedness of the implicit capital of a financial behaviour.

Notice that TTC can be looked upon as a special purpose timed process algebra in which processes are considered at a level of detail where the time slices in which actions are performed matter, but not their order within the time slices. This makes TTC suitable for formalizing financial products: financial products

Table 5. Axioms for delay, pre-abstraction and interest counting encapsulation

$\sigma(\epsilon) = \epsilon$		D1			
$\sigma(\delta) = \delta$		D2			
$\sigma(\gamma(u)) = \gamma(u)$		D3	$\gamma(1 - \frac{1+u}{1+u}) \oplus \partial^u_{\{a\}}(a(v) \oplus x) =$		
$\sigma(x \oplus y) = \sigma(x) \oplus \sigma(y)$		D4	$\gamma(1 - \frac{1+u}{1+u}) \oplus \partial^u_{\{a\}}(\sigma(a((1+u) \cdot v)) \oplus x)$		RICA
$\mathsf{t}_I(\epsilon) = \epsilon$		PA1	$\partial^u_H(\epsilon) = \epsilon$		ICE1
$\mathsf{t}_I(\delta) = \delta$		PA2	$\partial^u_H(\delta) = \delta$		ICE2
$\mathsf{t}_I(\gamma(u)) = \gamma(u)$		PA3	$\partial^u_H(\gamma(v)) = \gamma(v)$		ICE3
$\mathsf{t}_I(a(u)) = a(u)$	if $a \notin I$	PA4	$\partial^u_H(a(v)) = a(v)$	if $a \notin H$	ICE4
$\mathsf{t}_I(a(u)) = \mathsf{t}(u)$	if $a \in I$	PA5	$\partial^u_H(a(v)) = \gamma(v)$	if $a \in H$	ICE5
$\mathsf{t}_I(x \oplus y) = \mathsf{t}_I(x) \oplus \mathsf{t}_I(y)$		PA6	$\partial^u_H(x \oplus \partial^u_H(y)) = \partial^u_H(x) \oplus \partial^u_H(y)$		ICE6
$\mathsf{t}_I(\sigma(x)) = \sigma(\mathsf{t}_I(x))$		PA7	$\partial^u_H(\sigma(x)) = \sigma(\partial^u_H(x))$		ICE7
$\mathsf{t}_{I \cup I'}(x) = \mathsf{t}_I(\mathsf{t}_{I'}(x))$		PA8	$\partial^u_{H \cup H'}(x) = \partial^u_H(\partial^u_{H'}(x))$		ICE8

Table 6. Axioms for implicit capital

$\mathrm{Q}^u(x) = \mathrm{Q}^u(\mathsf{t}_A(x))$	IC1
$\mathrm{Q}^u(\epsilon) = 0$	IC2
$\mathrm{Q}^u(\delta) = -1$	IC3
$\mathrm{Q}^u(\mathsf{t}(v)) = \max(v, 0)$	IC4
$\frac{1+\mathrm{Q}^u(x)}{1+\mathrm{Q}^u(x)} \cdot \mathrm{Q}^u(\sigma(x)) = \frac{1+\mathrm{Q}^u(x)}{1+\mathrm{Q}^u(x)} \cdot \max(\frac{1}{1+u} \cdot \mathrm{Q}^u(x), 0)$	IC5
$\frac{1+\mathrm{Q}^u(x)}{1+\mathrm{Q}^u(x)} \cdot \mathrm{Q}^u(\mathsf{t}(v) \oplus \sigma(x)) = \frac{1+\mathrm{Q}^u(x)}{1+\mathrm{Q}^u(x)} \cdot \max(v + \frac{1}{1+u} \cdot \mathrm{Q}^u(x), 0)$	IC6

exhibit financial behaviours where the day, week, month or year in which actions are performed and the amounts of money are transferred in doing so are relevant, but not their order within the periods concerned.

The axioms of TTC are the axioms of CTC and the additional axioms given in Tables 5 and 6. Like in CTC, the proof rule DE is adopted to lift the valid equations between terms of sort **Q** to TTC.

Axiom RICA (Realistic Interest Calculation Axiom) is equivalent to

$$u \neq -1 \Rightarrow \partial^u_{\{a\}}(a(v) \oplus x) = \partial^u_{\{a\}}(\sigma(a((1+u) \cdot v)) \oplus x) \ ,$$

which can be paraphrased as follows: when encapsulating a, reckoning with an interest rate u different from -1, a transfer of an amount v in time slice n is equivalent to a transfer of an amount $(u+1) \cdot v$ in time slice $n+1$. The exclusion of $u = -1$ prevents that the equation $x = \delta$ can be derived. Axioms IC5 and IC6 are equivalent to

$$\mathrm{Q}^u(x) \neq -1 \Rightarrow \mathrm{Q}^u(\sigma(x)) = \max(\tfrac{1}{1+u} \cdot \mathrm{Q}^u(x), 0) \ ,$$

$$\mathrm{Q}^u(x) \neq -1 \Rightarrow \mathrm{Q}^u(\mathsf{t}(v) \oplus \sigma(x)) = \max(v + \tfrac{1}{1+u} \cdot \mathrm{Q}^u(x), 0) \ .$$

The exclusion of $\mathrm{Q}^u(x) = -1$ is needed because -1 is used to represent undefinedness of the implicit capital of a financial behaviour.

Example 1. Let p be a closed term of sort $\mathbf{Q}$ such that $\mathcal{D} \models \frac{1+p}{1+p} = 1$. The following is a derivation from the axioms of TTC and the proof rule DE:

$$\begin{aligned}
&\partial^p_{\{a\}}(a(u) \oplus \sigma(a(5)) \oplus \sigma^2(b(u-7))) \\
&= \partial^p_{\{a\}}(a(u) \oplus a(\tfrac{5}{1+p}) \oplus \sigma^2(\partial^p_{\{a\}}(b(u-7)))) \\
&= \partial^p_{\{a\}}(a(u+\tfrac{5}{1+p}) \oplus \partial^p_{\{a\}}(\sigma^2(b(u-7)))) \\
&= \partial^p_{\{a\}}(a(u+\tfrac{5}{1+p})) \oplus \partial^p_{\{a\}}(\sigma^2(b(u-7))) \\
&= \partial^p_{\{a\}}(a(u+\tfrac{5}{1+p})) \oplus \sigma^2(\partial^p_{\{a\}}(b(u-7))) \\
&= \gamma(u+\tfrac{5}{1+p}) \oplus \sigma^2(b(u-7)) \ .
\end{aligned}$$

Because $\mathcal{D} \models \frac{-5}{1+p} + \frac{5}{1+p} = 0$, it follows immediately that

$$\partial^p_{\{a\}}(a(\tfrac{-5}{1+p}) \oplus \sigma(a(5)) \oplus \sigma^2(b(\tfrac{-5}{1+p} - 7))) = \sigma^2(b(\tfrac{-5}{1+p} - 7)) \ .$$

Moreover, it follows immediately that

$$\partial^p_{\{a\}}(a(q) \oplus \sigma(a(5)) \oplus \sigma^2(b(q-7))) = \delta$$

for all closed terms q of sort $\mathbf{Q}$ such that not $\mathcal{D} \models q + \frac{5}{1+p} = 0$.

Example 2. Let p and q be closed terms of sort $\mathbf{Q}$. The following is a derivation from the axioms of TTC and the proof rule DE:

$$\begin{aligned}
&\mathrm{Q}^p(a(7) \oplus \sigma(a'(-8)) \oplus b(-5) \oplus \sigma^2(b'((1+q)^2 \cdot 5))) \\
&= \mathrm{Q}^p(\mathsf{t}(7) \oplus \sigma(\mathsf{t}(-8)) \oplus \mathsf{t}(-5) \oplus \sigma^2(\mathsf{t}((1+q)^2 \cdot 5))) \\
&= \mathrm{Q}^p(\mathsf{t}(2) \oplus \sigma(\mathsf{t}(-8) \oplus \sigma(\mathsf{t}((1+q)^2 \cdot 5)))) \\
&= \max(2 + \tfrac{1}{1+p} \cdot \mathrm{Q}^p(\mathsf{t}(-8) \oplus \sigma(\mathsf{t}((1+q)^2 \cdot 5))), 0) \\
&= \max(2 + \tfrac{1}{1+p} \cdot \max(-8 + \tfrac{1}{1+p} \cdot \mathrm{Q}^p(\mathsf{t}((1+q)^2 \cdot 5)), 0), 0) \\
&= \max(2 + \tfrac{1}{1+p} \cdot \max(-8 + \tfrac{1}{1+p} \cdot (1+q)^2 \cdot 5, 0), 0) \ .
\end{aligned}$$

It follows immediately that

$$\mathrm{Q}^p(a(7) \oplus \sigma(a'(-8)) \oplus b(-5) \oplus \sigma^2(b'((1+q)^2 \cdot 5))) = 2$$

for all closed terms p and q of sort $\mathbf{Q}$ such that $\mathcal{D} \models \frac{1}{1+p} \cdot (1+q)^2 \leq \frac{8}{5}$. There are many such p and q, for example, p and q such that $\mathcal{D} \models p = \frac{1}{100}$ and $\mathcal{D} \models q = \frac{10}{100}$, but also p and q such that $\mathcal{D} \models p = \frac{25}{100}$ and $\mathcal{D} \models q = \frac{40}{100}$. We will return to this example in Section 5, where p and q are taken as interest rates on savings and loans, respectively.

To prove a statement for all tuplix-closed TTC terms of sort $\mathbf{T}$, it is sufficient to prove it for all tuplix-closed TTC canonical terms. The set of *TTC canonical terms* is inductively defined by the following rules:

– if t is a CTC canonical term, then t is a TTC canonical term;
– if t is a CTC canonical term and t' is a TTC canonical term, then $t \mathbin{⊙} \sigma(t')$ is a TTC canonical term.

Proposition 2. *For all tuplix-closed* TTC *terms t of sort* $\mathbf{T}$*, there exists a tuplix-closed* TTC *canonical term t' such that $t = t'$ is derivable from the axioms of* TTC*.*

Proof. The proof is straightforward by induction on the structure of t, and in the cases $t \equiv \mathtt{t}_I(s)$ and $t \equiv \partial^p_H(s)$ (where we can restrict ourselves to tuplix-closed TTC canonical terms s) by induction on the structure of s. The following easy to prove fact is used in the proof for the case $t \equiv \partial^p_H(s)$: for all TTC terms t_1 of sort $\mathbf{T}$ and all tuplix-closed TTC terms t_2 of sort $\mathbf{T}$ in which no element of H occurs, $\partial^u_H(t_1 \mathbin{⊙} t_2) = \partial^u_H(t_1) \mathbin{⊙} t_2$ is derivable from the axioms of TTC (cf. Lemma 5 in [6]). □

5 Pure Financial Products

TTC is concerned with financial behaviours. A financial product exhibits a financial behaviour. In [10], Wesseling and van den Bergh propose a cumulative interest compliant conservation requirement as a criterion for pure financial products: the sum of all transfers relating to the product, transposed to some point of time by means of cumulative interest at the actual interest rate of the product, is zero. In this section we formalize this requirement and show how financial behaviours may profit from using pure financial products.

Let p be a closed term of sort $\mathbf{Q}$. Then the cumulative interest compliant conservation requirement for interest rate p is formalized by the equation

$$\partial^p_{\{\mathtt{t}\}}(\mathtt{t}_A(x)) = \epsilon \ .$$

This equation is called the *Wesseling and van den Bergh equation* or shortly the *W-vdB equation*. Suppose that p represents the actual interest rate of a financial product. Then a tuplix that represents the behaviour of that financial product is a tuplix that represent the behaviour of a *pure financial product* if it satisfies the W-vdB equation for interest rate p.

Let p be a closed term of sort $\mathbf{Q}$ and t be a closed term of sort $\mathbf{T}$. Suppose that p represents the actual interest rate of a pure financial product and t represents the behaviour of that pure financial product. If that pure financial product is a financial product of credit type, then usually $\mathrm{Q}^p(t) = 0$. However, if that pure financial product is a financial product of savings type, then $\mathrm{Q}^p(t) > 0$.

Let p and q be closed terms of sort $\mathbf{Q}$ and let t and t' be closed terms of sort $\mathbf{T}$. Suppose that p represents the expected interest rate on savings and q represents the actual interest rate of a pure financial product of credit type. Moreover, suppose that t represents the behaviour of that pure financial product and t' represents some wanted financial behaviour. We say that the wanted financial behaviour t' profits from using the pure financial product whose behaviour is

t with interest rate p on savings if $\mathrm{Q}^p(t \mathbin{\textcircled{\scriptsize :}} t') < \mathrm{Q}^p(t')$. In any case, we have $\mathrm{Q}^p(t \mathbin{\textcircled{\scriptsize :}} t') \leq \mathrm{Q}^p(t) + \mathrm{Q}^p(t')$. The important observation is that we may have $\mathrm{Q}^p(t \mathbin{\textcircled{\scriptsize :}} t') < \mathrm{Q}^p(t')$. As an example, we take the case where p and q are such that $\mathcal{D} \models \frac{1}{1+p} \cdot (1+q)^2 \leq \frac{8}{5}$, $t \equiv b(-5) \mathbin{\textcircled{\scriptsize :}} \sigma^2(b'((1+q)^2 \cdot 5))$, and $t' \equiv a(7) \mathbin{\textcircled{\scriptsize :}} \sigma(a'(-8))$. We have that $\partial^p_{\{\mathsf{t}\}}(\mathsf{t}_A(t)) = \epsilon$, so t actually represents the behaviour of a pure financial product. We can easily derive that $\mathrm{Q}^p(t) = 0$ and $\mathrm{Q}^p(t') = 7$. In Example 2, we have already derived that $\mathrm{Q}^p(t \mathbin{\textcircled{\scriptsize :}} t') = 2$. This means that in this case $\mathrm{Q}^p(t \mathbin{\textcircled{\scriptsize :}} t') < \mathrm{Q}^p(t')$. In other words, the wanted financial behaviour t' profits from the pure financial product t with interest rate p on savings.

Let p be a closed term of sort $\mathbf{Q}$, and let t' be a closed term of sort $\mathbf{T}$. Suppose that p represents the expected interest rate on savings and t' represents some wanted financial behaviour. Then there exists a closed term t of sort $\mathbf{T}$ that represents the behaviour of a pure financial product of credit type such that $\mathrm{Q}^p(t) = 0$ and $\mathrm{Q}^p(t \mathbin{\textcircled{\scriptsize :}} t') = 0$:

- if $\mathrm{Q}^p(t') = 0$, then take $t \equiv \epsilon$;
- if $\mathrm{Q}^p(t') > 0$, then take $t \equiv a(-\mathrm{Q}^p(t')) \mathbin{\textcircled{\scriptsize :}} \sigma^n(a'((1+p)^n \cdot \mathrm{Q}^p(t'))))$, where n is the greatest number of nested occurrences of terms of the form $\sigma(t'')$ in t'.

In other words, we arrive at the conclusion that any wanted financial behaviour can be exhibited without capital by using a pure financial product of credit type if we take the actual interest rate of the pure financial product as the expected interest rate on savings.

6 Standard Model of TTC

In this section, we construct the standard model of TTC. The standard model of CTC presented in [6] lies at the root of this model. However, the use of partial functions is circumvented.

We write $\mathcal{D}$ for the domain of the signed cancellation meadow $\mathcal{D}$, and we write $\Diamond$, where $\Diamond$ is a constant or operator from the signature of signed cancellation meadows, for the interpretation of $\Diamond$ in $\mathcal{D}$. To prevent confusion, we write $\underline{0}$ and $\underline{1}$ for the identity elements of addition and multiplication on natural numbers.

We define the set $\mathcal{TE}$ of *tuplix elements*, the set $\mathcal{UT}$ of *untimed tuplices*, and the set $\mathcal{TT}$ of *timed tuplices* as follows:

$$\mathcal{TE} = \bigcup_{A' \subseteq A} (A' \to \mathcal{D}) \,,$$

$$\mathcal{UT} = \{U \subseteq \mathcal{TE} \mid \mathrm{card}(U) \leq \underline{1}\} \,,$$

$$\mathcal{TT} = \{T : \mathbb{N} \to \mathcal{UT} \mid \forall i \in \mathbb{N} \bullet \mathrm{card}(T(i)) = \underline{0} \vee \forall i \in \mathbb{N} \bullet \mathrm{card}(T(i)) = \underline{1}\} \,.$$

In the definition of the standard model of TTC, the auxiliary set $\mathcal{TT}^-$ defined by

$$\mathcal{TT}^- = \{T \in \mathcal{TT} \mid \forall i \in \mathbb{N} \bullet \mathrm{card}(T(i)) = \underline{1}\}$$

is used as well. We write $el(U)$, where $U \in \mathcal{UT}$, for the unique element $f \in \mathcal{TE}$ such that $f \in U$ if $\mathrm{card}(U) = \underline{1}$, and an arbitrary $f \in \mathcal{TE}$ otherwise.

Table 7. Interpretation of constants and operators of TTC

$$
\begin{array}{lll}
\epsilon(i) & = \{[\,]\} & \\
\delta(i) & = \emptyset & \\
a(d)(i) & = \begin{cases} \{[a \mapsto d]\} & \text{if } i = \underline{0} \\ \{[\,]\} & \text{otherwise} \end{cases} & \\
\gamma(d)(i) & = \begin{cases} \{[\,]\} & \text{if } d = 0 \\ \emptyset & \text{otherwise} \end{cases} & \\
(T \mathbin{\hat{\oplus}} T')(i) & = \{f \mathbin{\hat{\oplus}} f' \mid f \in T(i) \wedge f' \in T'(i)\} & \\
\sigma(T)(i) & = \begin{cases} T(i - \underline{1}) & \text{if } i > \underline{0} \wedge T(i) \neq \emptyset \\ \{[\,]\} & \text{if } i = \underline{0} \wedge T(i) \neq \emptyset \\ \emptyset & \text{otherwise} \end{cases} & \\
\mathtt{t}_I(T)(i) & = \{\hat{\mathtt{t}}_I(f) \mid f \in T(i)\} & \\
\partial^d_H(T)(i) & = \{\hat{\epsilon}_H(f) \mid f \in T(i) \wedge \forall a \in H \bullet \mathrm{Total}^d_a(T) = 0\} & \\
\mathrm{Q}^d(T) & = \begin{cases} \hat{\mathrm{Q}}^d(T) & \text{if } \exists i \geq \underline{0} \bullet T(i) \neq \emptyset \\ -1 & \text{otherwise} \end{cases} &
\end{array}
$$

The *standard model* of TTC, written $\mathcal{M}(\mathcal{D}, A)$, is the expansion of the signed cancellation meadow $\mathcal{D}$ with

- for the sort $\mathbf{T}$, the set $\mathcal{TT}$;
- for each additional constant $\Diamond_0 : \mathbf{T}$ of TTC, the element $\Diamond_0 \in \mathcal{TT}$ defined in Table 7;
- for each additional operator $\Diamond_n : S_1 \times \ldots \times S_n \to S_{n+1}$ of TTC, the operation $\Diamond_n : D_1 \times \ldots D_n \to D_{n+1}$, where $D_i = \mathcal{TT}$ if $S_i \equiv \mathbf{T}$ and $D_i = \mathcal{D}$ if $S_i \equiv \mathbf{Q}$, defined in Table 7.[1]

In Table 7, the following auxiliary functions are used:

- the function $\hat{\oplus} : \mathcal{TE} \times \mathcal{TE} \to \mathcal{TE}$ defined by
  - $\mathrm{dom}(f \mathbin{\hat{\oplus}} f') = \mathrm{dom}(f) \cup \mathrm{dom}(f')$;
  - for each $a \in \mathrm{dom}(f \mathbin{\hat{\oplus}} f')$:
$$(f \mathbin{\hat{\oplus}} f')(a) = \begin{cases} f(a) + f'(a) & \text{if } a \in \mathrm{dom}(f) \cap \mathrm{dom}(f') \\ f(a) & \text{if } a \in \mathrm{dom}(f) \setminus \mathrm{dom}(f') \\ f'(a) & \text{if } a \in \mathrm{dom}(f') \setminus \mathrm{dom}(f)\ ; \end{cases}$$
- for each $I \subseteq A$, the function $\hat{\mathtt{t}}_I : \mathcal{TE} \to \mathcal{TE}$ defined by
  - $\mathrm{dom}(\hat{\mathtt{t}}_I(f)) = (\mathrm{dom}(f) \setminus I) \cup \{\mathtt{t} \mid \mathrm{dom}(f) \cap I \neq \emptyset\}$;
  - for each $a \in \mathrm{dom}(\hat{\mathtt{t}}_I(f))$:
$$\hat{\mathtt{t}}_I(f)(a) = \begin{cases} f(a) & \text{if } a \neq \mathtt{t} \\ \sum_{a' \in I} f(a') & \text{if } a = \mathtt{t}\ ; \end{cases}$$

[1] We write $[\,]$ for the empty function and $[e \mapsto e']$ for the function f with $\mathrm{dom}(f) = \{e\}$ such that $f(e) = e'$.

– for each $H \subseteq A$, the function $\hat{\epsilon}_H : \mathcal{TE} \to \mathcal{TE}$ defined by
  - $\mathrm{dom}(\hat{\epsilon}_H(f)) = \mathrm{dom}(f) \setminus H$;
  - for each $a \in \mathrm{dom}(\hat{\epsilon}_H(f))$:
    $\hat{\epsilon}_H(f)(a) = f(a)$;
– for each $a \in A$, the function $\mathrm{Total}_a : \mathcal{D} \times \mathcal{TT} \to \mathcal{D}$ defined by

$$\mathrm{Total}_a^d(T) = \sum_{i \text{ s.t. } a \in \mathrm{dom}(el(T(i)))} (1+d)^i \cdot el(T(i))(a) \; ;$$

– the function $\hat{\mathrm{Q}} : \mathcal{D} \times \mathcal{TT}^- \to \mathcal{D}$ defined by

$$\hat{\mathrm{Q}}^u(T) = \begin{cases} \max(q_0(T), 0) & \text{if } \forall i > \underline{0} \bullet T(i) = \{[]\} \\ \max(q_0(T) + \frac{1}{1+u} \cdot \hat{\mathrm{Q}}^u(sh(T)), 0) & \text{if } \exists i > \underline{0} \bullet T(i) \neq \{[]\} \end{cases} ,$$

where:
- $sh : \mathcal{TT}^- \to \mathcal{TT}^-$ is defined by $sh(T)(i) = T(i + \underline{1})$ for all $i \in \mathbb{N}$;
- $q_0 : \mathcal{TT}^- \to \mathcal{D}$ is defined by $q_0(T) = \sum_{a \in \mathrm{dom}(el(T(\underline{0})))} el(T(\underline{0}))(a)$.

It is easy to establish the following soundness result: for all terms t and t' of sort $\mathbf{T}$, $t = t'$ is derivable from the axioms of TTC and the proof rule DE only if $\mathcal{M}(\mathcal{D}, A) \models t = t'$. We also have a completeness result.

Theorem 1. *For all closed terms t and t' of sort $\mathbf{T}$, $\mathcal{M}(\mathcal{D}, A) \models t = t'$ only if $t = t'$ is derivable from the axioms of* TTC *and the proof rule* DE.

Proof. By Proposition 2, it is sufficient to show that, for all closed TTC canonical terms t and t', $\mathcal{M}(\mathcal{D}, A) \models t = t'$ only if $t = t'$ is derivable from the axioms of TTC and the proof rule DE. This is easy to prove by induction on the structure of t using Theorem 1 from [6]. □

7 Conclusions

We have developed a timed extension of tuplix calculus. The result can be looked upon as a special purpose timed process algebra in which processes are considered at a level of detail where the time slices in which actions are performed matter, but not their order within the time slices. This makes it suited for the description and analysis of financial products: financial products exhibit financial behaviours where the day, week, month or year in which actions are performed and the amounts of money are transferred in doing so are relevant, but not their order within the periods concerned.

We have formalized the cumulative interest compliant conservation requirement for pure financial products proposed by Wesseling and van den Bergh by an equation in the timed tuplix calculus developed. Thus, a formalization of the starting-point of the material on the mathematics of finance presented in [10] has been achieved. Moreover, we have used the timed tuplix calculus developed to show how wanted financial behaviours may profit from using pure financial

products. The timed tuplix calculus appears to be a reasonable setting for further work in this area.

Like Wesseling and van den Bergh, we consider only financial products of which the interest rate is not dependent on changes in the financial market. If the interest rate of a financial product is made dependent on changes in the financial market, then the expressiveness of the timed tuplix calculus is insufficient. In this more dynamic case, discrete time process algebra [2] looks to be a reasonable setting for the formalization of an adapted version of the cumulative interest compliant conservation requirement.

Wesseling and van den Bergh have informed us that their conservation requirement for pure financial products has been formulated under the influence of basic ideas on the mathematics of finance presented in [8].

References

1. Baeten, J.C.M., Bergstra, J.A.: Global renaming operators in concrete process algebra. Information and Control **78**(3), 205–245 (1988)
2. Baeten, J.C.M., Bergstra, J.A.: Discrete time process algebra. Formal Aspects of Computing **8**(2), 188–208 (1996)
3. Bergstra, J.A., Klop, J.W.: Process algebra for synchronous communication. Information and Control **60**(1/3), 109–137 (1984)
4. Bergstra, J.A., Middelburg, C.A.: Parallel processes with implicit computational capital. Electronic Notes in Theoretical Computer Science **209**, 55–81 (2008)
5. Bergstra, J.A., Ponse, A.: A generic basis theorem for cancellation meadows. `arXiv:0803.3969v2 [math.RA]` at `http://arxiv.org/` (2008)
6. Bergstra, J.A., Ponse, A., van der Zwaag, M.B.: Tuplix calculus. `arXiv:0712.3423v1 [cs.LO]` at `http://arxiv.org/` (2007)
7. Bergstra, J.A., Tucker, J.V.: The rational numbers as an abstract data type. Journal of the ACM **54**(2), Article 7 (2007)
8. Cissell, R., Cissell, H., Flaspohler, D.: Mathematics of Finance. Houghton Mifflin, Boston (1990)
9. Sannella, D., Tarlecki, A.: Algebraic preliminaries. In: E. Astesiano, H.J. Kreowski, B. Krieg-Brückner (eds.) Algebraic Foundations of Systems Specification, pp. 13–30. Springer-Verlag, Berlin (1999)
10. Wesseling, J., van den Bergh, A.: Realistische Interestberekeningen. Academic Service, Schoonhoven, the Netherlands (2000)
11. Wirsing, M.: Algebraic specification. In: J. van Leeuwen (ed.) Handbook of Theoretical Computer Science, vol. B, pp. 675–788. Elsevier, Amsterdam (1990)

Electronic Reports Series of the Programming Research Group

Within this series the following reports appeared.

[PRG0814] J.A. Bergstra and C.A. Middelburg, *Instruction Sequences for the Production of Processes*, Programming Research Group - University of Amsterdam, 2008.

[PRG0813] J.A. Bergstra and C.A. Middelburg, *On the Expressiveness of Single-Pass Instruction Sequences*, Programming Research Group - University of Amsterdam, 2008.

[PRG0812] J.A. Bergstra and C.A. Middelburg, *Instruction Sequences and Non-uniform Complexity Theory*, Programming Research Group - University of Amsterdam, 2008.

[PRG0811] D. Staudt, *A Case Study in Software Engineering with PSF: A Domotics Application*, Programming Research Group - University of Amsterdam, 2008.

[PRG0810] J.A. Bergstra and C.A. Middelburg, *Thread Algebra for Poly-Threading*, Programming Research Group - University of Amsterdam, 2008.

[PRG0809] J.A. Bergstra and C.A. Middelburg, *Data Linkage Dynamics with Shedding*, Programming Research Group - University of Amsterdam, 2008.

[PRG0808] B. Diertens, *A Process Algebra Software Engineering Environment*, Programming Research Group - University of Amsterdam, 2008.

[PRG0807] J.A. Bergstra, S. Nolst Trenite, and M.B. van der Zwaag, *Tuplix Calculus Specifications of Financial Transfer Networks*, Programming Research Group - University of Amsterdam, 2008.

[PRG0806] J.A. Bergstra and C.A. Middelburg, *Data Linkage Algebra, Data Linkage Dynamics, and Priority Rewriting*, Programming Research Group - University of Amsterdam, 2008.

[PRG0805] J.A. Bergstra, S. Nolst Trenite, and M.B. van der Zwaag, *UvA Budget Allocatie Model*, Programming Research Group - University of Amsterdam, 2008.

[PRG0804] J.A. Bergstra and C.A. Middelburg, *Thread Algebra for Sequential Poly-Threading*, Programming Research Group - University of Amsterdam, 2008.

[PRG0803] J.A. Bergstra and C.A. Middelburg, *Thread Extraction for Polyadic Instruction Sequences*, Programming Research Group - University of Amsterdam, 2008.

[PRG0802] A. Barros and T. Hou, *A Constructive Version of AIP Revisited*, Programming Research Group - University of Amsterdam, 2008.

[PRG0801] J.A. Bergstra and C.A. Middelburg, *Programming an Interpreter Using Molecular Dynamics*, Programming Research Group - University of Amsterdam, 2008.

[PRG0713] J.A. Bergstra, A. Ponse, and M.B. van der Zwaag, *Tuplix Calculus*, Programming Research Group - University of Amsterdam, 2007.

[PRG0712] J.A. Bergstra, S. Nolst Trenite, and M.B. van der Zwaag, *Towards a Formalization of Budgets*, Programming Research Group - University of Amsterdam, 2007.

[PRG0711] J.A. Bergstra and C.A. Middelburg, *Program Algebra with a Jump-Shift Instruction*, Programming Research Group - University of Amsterdam, 2007.

[PRG0710] J.A. Bergstra and C.A. Middelburg, *Instruction Sequences with Dynamically Instantiated Instructions*, Programming Research Group - University of Amsterdam, 2007.

[PRG0709] J.A. Bergstra and C.A. Middelburg, *Instruction Sequences with Indirect Jumps*, Programming Research Group - University of Amsterdam, 2007.

[PRG0708] B. Diertens, *Software (Re-)Engineering with PSF III: an IDE for PSF*, Programming Research Group - University of Amsterdam, 2007.

[PRG0707] J.A. Bergstra and C.A. Middelburg, *An Interface Group for Process Components*, Programming Research Group - University of Amsterdam, 2007.

[PRG0706] J.A. Bergstra, Y. Hirschfeld, and J.V. Tucker, *Skew Meadows*, Programming Research Group - University of Amsterdam, 2007.

[PRG0705] J.A. Bergstra, Y. Hirschfeld, and J.V. Tucker, *Meadows*, Programming Research Group - University of Amsterdam, 2007.

[PRG0704] J.A. Bergstra and C.A. Middelburg, *Machine Structure Oriented Control Code Logic (Extended Version)*, Programming Research Group - University of Amsterdam, 2007.

[PRG0703] J.A. Bergstra and C.A. Middelburg, *On the Operating Unit Size of Load/Store Architectures*, Programming Research Group - University of Amsterdam, 2007.

[PRG0702] J.A. Bergstra and A. Ponse, *Interface Groups and Financial Transfer Architectures*, Programming Research Group - University of Amsterdam, 2007.

[PRG0701] J.A. Bergstra, I. Bethke, and M. Burgess, *A Process Algebra Based Framework for Promise Theory*, Programming Research Group - University of Amsterdam, 2007.

[PRG0610] J.A. Bergstra and C.A. Middelburg, *Parallel Processes with Implicit Computational Capital*, Programming Research Group - University of Amsterdam, 2006.

[PRG0609] B. Diertens, *Software (Re-)Engineering with PSF II: from architecture to implementation*, Programming Research Group - University of Amsterdam, 2006.

[PRG0608] A. Ponse and M.B. van der Zwaag, *Risk Assessment for One-Counter Threads*, Programming Research Group - University of Amsterdam, 2006.

[PRG0607] J.A. Bergstra and C.A. Middelburg, *Synchronous Cooperation for Explicit Multi-Threading*, Programming Research Group - University of Amsterdam, 2006.

[PRG0606] J.A. Bergstra and M. Burgess, *Local and Global Trust Based on the Concept of Promises*, Programming Research Group - University of Amsterdam, 2006.

[PRG0605] J.A. Bergstra and J.V. Tucker, *Division Safe Calculation in Totalised Fields*, Programming Research Group - University of Amsterdam, 2006.

[PRG0604] J.A. Bergstra and A. Ponse, *Projection Semantics for Rigid Loops*, Programming Research Group - University of Amsterdam, 2006.

[PRG0603] J.A. Bergstra and I. Bethke, *Predictable and Reliable Program Code: Virtual Machine-based Projection Semantics (submitted for inclusion in the Handbook of Network and Systems Administration)*, Programming Research Group - University of Amsterdam, 2006.

[PRG0602] J.A. Bergstra and A. Ponse, *Program Algebra with Repeat Instruction*, Programming Research Group - University of Amsterdam, 2006.

[PRG0601] J.A. Bergstra and A. Ponse, *Interface Groups for Analytic Execution Architectures*, Programming Research Group - University of Amsterdam, 2006.

[PRG0505] B. Diertens, *Software (Re-)Engineering with PSF*, Programming Research Group - University of Amsterdam, 2005.

[PRG0504] P.H. Rodenburg, *Piecewise Initial Algebra Semantics*, Programming Research Group - University of Amsterdam, 2005.

[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/