Discrete time process algebra with abstraction

Eindhoven University of Technology
Department of Mathematics and Computing Science

Discrete Time Process Algebra with Abstraction

by

J.C.M. Baeten and J.A. Bergstra

95/19

ISSN 0926-4515

Computing Science Report 95/19
Eindhoven, June 1995

Discrete Time Process Algebra with Abstraction

J.C.M. Baeten
Department of Computing Science, Eindhoven University of Technology, P.O.Box 513, 5600 MB Eindhoven, The Netherlands

J.A. Bergstra
Programming Research Group, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
and
Department of Philosophy, Utrecht University, Heidelberglaan 8, 3584 CS Utrecht, The Netherlands

The axiom system ACP of [BEK84a] was extended to discrete time in [BAB95]. Here we proceed to define the silent step in this theory in branching bisimulation semantics [GLW91, BAW90] rather than weak bisimulation semantics [MIL89, BEK85]. We present versions based on relative timing and on absolute timing. Both approaches are integrated using parametric timing. The time free ACP theory is embedded in the discrete time theory.
Note: Partial support received from ESPRIT Basic Research Action 7166, CONCUR2. To appear in the proceedings of FCT'95, Dresden.

1. INTRODUCTION.

Process algebra in the form of ACP [BEK85, BAW90], CCS [MIL89] or CSP [BRHR84] involves no explicit mention of time. Time is present in the interpretation of sequential composition: in p·q (ACP notation) the process p should be executed before q. Process algebras can be introduced that support standardised features to incorporate a quantative view on time. Time may be represented by means of non-negative reals, and actions can be given time stamps. This line is followed in [BAB91] for ACP, in [MOT90] for CCS and in [RER88] for CSP.

A second option is to divide time in slices indexed by natural numbers, to have an implicit or explicit time stamping mechanism that determines for each action the time slice in which it occurs and to have a time order within each slice only. This line has been followed in ATP [NSVR90], [NIS94], a process algebra that adds time slicing to a version of ACP based on action prefixing rather than sequential composition. Further, [GRO90] has extended ACP with time slices whereas [MOT89] have added these features to CCS. Following [BAB95], we use the phrase discrete time process algebra if an enumeration of time slices is used.

The objective of this paper is to extend the discrete time process algebra of ACP as given in [BAB95] with the silent step τ. This will allow a notion of abstraction. We base our work on the branching bisimulation of [GLW91]. We mention that [KLU93] has extended the real time ACP of [BAB91] with silent steps. We present three views on discrete time process algebra with silent step. In section 2, we consider discrete time process algebra with abstraction in *relative timing*, where timing refers to the execution of the previous action. In section 3, we have discrete time process algebra with abstraction in *absolute timing*, where all timing refers to an absolute clock. Here again, we only consider the two-phase version. In section 4, we have discrete time process algebra with *parametric timing*, where absolute and relative timing are integrated. For parametric timing, we introduce a model based on time spectrum sequences.

An underlying viewpoint of the present paper is that for a given time free atomic action, there may be different timed versions. We mention some: $\mathrm{fts}(a)$ stands for action a in the first time slice, followed by immediate termination, $\mathrm{cts}(a)$ is a in the current time slice with immediate termination, $\mathrm{ats}(a)$ is a in any time slice with immediate termination, $\mathrm{atstau}(a) = \mathrm{ats}(a)\cdot\mathrm{ats}(\tau)$ is a in any time slice followed by silent termination in any subsequent slice. It turns out that for a an atomic action or τ, the interpretation of a as $\mathrm{atstau}(a)$ is the appropriate (homomorphic) embedding of time free process algebra into timed process algebra.

There are many practical uses conceivable for timed process algebras. In particular, we mention the TOOLBUS (see [BEK94, 95]). This TOOLBUS contains a program notation called T which is syntactically sugared discrete time process algebra. Programs in T are called T-scripts. The runtime system is also described in terms of discrete time process algebra. By using randomised symbolic execution the TOOLBUS implementation enacts that the axioms of process algebra can be viewed as correctness preserving transformations of T-scripts. A comparable part of disrete time process algebra that is used to describe T-scripts has also been used for the description of ϕSDL, flat SDL, a subset of SDL that leaves out modularisation and concentrates on timing aspects (see [BEM95]).

2. DISCRETE TIME PROCESS ALGEBRA WITH RELATIVE TIMING.

We start out from the relative discrete time process algebra of [BAB95]. First, we consider the theory with only nondelayable actions, next we add the delayable or time free actions.

2.1 BASIC PROCESS ALGEBRA WITHOUT TIME FREE ACTIONS.

The signature of $\mathrm{BPA}_{\mathrm{drt}}^{-}$ has constants $\mathrm{cts}(a)$ (for $a \in A$), denoting a in the current time slice, and $\mathrm{cts}(\delta)$, denoting a deadlock at the end of the current time slice. The superscript $^{-}$ denotes that time free atoms are not part of the signature. Also, we have the immediate deadlock constant $\dot{\delta}$ introduced in [BAB95]. This constant denotes an immediate and catastrophic deadlock. Within a time slice, there is no explicit mention of the passage of time, we can see the passage to the next time slice as a clock tick. Thus, the $\mathrm{cts}(a)$ can be called nondelayable actions: the action must occur before the next clock tick. The operators are alternative and sequential composition, and the relative discrete time unit delay σ_{rel} (the notation σ taken from [HER90]). The process $\sigma_{\mathrm{rel}}(x)$ will start x after one clock tick, i.e. in the next time slice. In addition, we add the auxiliary operator ν_{rel}. This operator disallows an initial time step, it gives the part of a process that starts with an action in the current time slice. It was called a time out at the end of the current time slice in [BAB92], there, the notation $x \gg_{\mathrm{dt}} 1$ was used for $\nu_{\mathrm{rel}}(x)$. (The greek letter ν sounds like "now"; this correspondence is even stronger in Dutch.)

The axiom DRT1 is the time factorization axiom: it says that the passage of time by itself cannot determine a choice. The addition of a silent step in strong bisimulation semantics now just amounts to the presence of a new constant $\mathrm{cts}(\tau)$, with the same axioms as the $\mathrm{cts}(a)$ constants. We write $A_\tau = A\cup\{\tau\}, A_\delta = A\cup\{\delta\}$ etc.

The standard process algebra BPA_δ can be considered as an SRM specification (Subalgebra of Reduced Model, in the terminology of [BAB94]) of the present theory: consider the initial algebra of $\mathrm{BPA}_{\mathrm{drt}}^{-} + \mathrm{DCS}$, reduce the signature by omitting $\dot{\delta}$, σ_{rel}, ν_{rel}, then BPA_δ is a complete axiomatisation of the reduced model, under the interpretation of a,δ by $\mathrm{cts}(a)$, $\mathrm{cts}(\delta)$ (note that $x + \mathrm{cts}(\delta) = x$ for all closed terms x except $\dot{\delta}$).

$x + y = y + x$	A1	$\sigma_{\mathrm{rel}}(x) + \sigma_{\mathrm{rel}}(y) = \sigma_{\mathrm{rel}}(x + y)$	DRT1
$(x + y) + z = x + (y + z)$	A2	$\sigma_{\mathrm{rel}}(x)\cdot y = \sigma_{\mathrm{rel}}(x\cdot y)$	DRT2
$x + x = x$	A3	$\sigma_{\mathrm{rel}}(\dot{\delta}) = \mathrm{cts}(\delta)$	DRT3
$(x + y)\cdot z = x\cdot z + y\cdot z$	A4	$\mathrm{cts}(a) + \mathrm{cts}(\delta) = \mathrm{cts}(a)$	DRT4
$(x\cdot y)\cdot z = x\cdot(y\cdot z)$	A5	$\nu_{\mathrm{rel}}(\dot{\delta}) = \dot{\delta}$	DCS1
		$\nu_{\mathrm{rel}}(\mathrm{cts}(a)) = \mathrm{cts}(a)$	DCS2
$x + \dot{\delta} = x$	A6ID	$\nu_{\mathrm{rel}}(\sigma_{\mathrm{rel}}(x)) = \mathrm{cts}(\delta)$	DCS3
$\dot{\delta}\cdot x = \dot{\delta}$	A7ID	$\nu_{\mathrm{rel}}(x + y) = \nu_{\mathrm{rel}}(x) + \nu_{\mathrm{rel}}(y)$	DCS4
		$\nu_{\mathrm{rel}}(x\cdot y) = \nu_{\mathrm{rel}}(x)\cdot y$	DCS5

TABLE 1. $\mathrm{BPA}_{\mathrm{drt}}^{-} + \mathrm{DCS}$.

2.2 STRUCTURED OPERATIONAL SEMANTICS.

We give a semantics in terms of operational rules. We have the following relations on the set of closed process expressions P:

action step $\subseteq P \times A_\tau \times P$, notation $p \xrightarrow{a} p'$	(denotes action execution)
action termination $\subseteq S \times A_\tau$, notation $p \xrightarrow{a} \surd$	(execution of a terminating action).
time step $\subseteq P \times P$, notation $p \xrightarrow{\sigma} p'$	(denotes passage to the next time slice)
immediate deadlock $\subseteq P$, notation $\mathrm{ID}(p)$	(immediate deadlock, holds only for process expressions equal to $\dot{\delta}$).

We enforce the time factorization axiom DRT1 by phrasing the rules so that each process expression has at most one σ-step: in a transition system, each node has at most one outgoing σ-edge. This operational semantics uses predicates and negative premises. Still, using terminology and results of [VER94], the rules satisfy the *panth* format, and determine a unique transition relation on closed process expressions. Strong bisimulation is defined as usual, so a binary relation R on P is a *strong bisimulation* iff the following conditions hold:

i. if $R(p,q)$ and $p \xrightarrow{u} p'$ $(u \in A_{\tau\sigma})$, then there is q' such that $q \xrightarrow{u} q'$ and $R(p',q')$

ii. if $R(p,q)$ and $q \xrightarrow{u} q'$ $(u \in A_{\tau\sigma})$, then there is p' such that $p \xrightarrow{u} p'$ and $R(p',q')$

iii. if $R(p,q)$ then $p \xrightarrow{a} \surd$ iff $q \xrightarrow{a} \surd$ $(a \in A_\tau)$ and $\mathrm{ID}(p)$ iff $\mathrm{ID}(q)$.

Two terms p,q are *strong bisimulation equivalent*, $p \underline{\leftrightarrow} q$, if there exists a strong bisimulation relating them. From [BAB95] we know that the axiomatisation in table 1 is sound and complete for the model of closed process expressions modulo strong bisimulation.

$$\mathrm{cts}(a) \xrightarrow{a} \surd \qquad \mathrm{ID}(\dot{\delta}) \qquad \frac{\neg\,\mathrm{ID}(x)}{\sigma_{\mathrm{rel}}(x) \xrightarrow{\sigma} x}$$

$$\frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y} \qquad \frac{x \xrightarrow{a} \surd}{x \cdot y \xrightarrow{a} y} \qquad \frac{x \xrightarrow{\sigma} x'}{x \cdot y \xrightarrow{\sigma} x' \cdot y} \qquad \frac{\mathrm{ID}(x)}{\mathrm{ID}(x \cdot y)}$$

$$\frac{x \xrightarrow{a} x'}{x+y \xrightarrow{a} x',\ y+x \xrightarrow{a} x'} \qquad \frac{x \xrightarrow{a} \surd}{x+y \xrightarrow{a} \surd,\ y+x \xrightarrow{a} \surd} \qquad \frac{\mathrm{ID}(x),\ \mathrm{ID}(y)}{\mathrm{ID}(x+y)}$$

$$\frac{x \xrightarrow{\sigma} x',\ y \xrightarrow{\sigma} y'}{x+y \xrightarrow{\sigma} x'+y'} \qquad \frac{x \xrightarrow{\sigma} x',\ y \not\xrightarrow{\sigma}}{x+y \xrightarrow{\sigma} x',\ y+x \xrightarrow{\sigma} x'}$$

$$\frac{x \xrightarrow{a} x'}{v_{\mathrm{rel}}(x) \xrightarrow{a} x'} \qquad \frac{x \xrightarrow{a} \surd}{v_{\mathrm{rel}}(x) \xrightarrow{a} \surd} \qquad \frac{\mathrm{ID}(x)}{\mathrm{ID}(v_{\mathrm{rel}}(x))}$$

TABLE 2. Operational rules for $\mathrm{BPA}_{\mathrm{d}\overline{\mathrm{r}}\mathrm{t}}$ + DCS ($a \in A_\tau$).

2.3 BRANCHING BISIMULATION.

Now, we want to define a notion of bisimulation that takes into account the special status of the silent step. We start from the definition of branching bisimulation in the time free case, and adapt this to our timed setting. An immediate deadlock term can only be related to another immediate deadlock term. As a consequence, we can have no term that is related to $\surd$. This is different from the usual definition of branching bisimulation of [GLW91, BAW90]. In order to emphasise this fact, we call the relations to be defined branching tail bisimulations. A useful notation is to write $p \Rightarrow q$ if it is possible to reach q from p by executing a number of τ-steps (0 or more).

A binary relation R on P is a *branching tail bisimulation* if:

i. if $R(p,q)$ and $p \xrightarrow{u} p'$ ($u \in A_{\tau\sigma}$) then either:
   a. $u \equiv \tau$ and $R(p',q)$, or
   b. there are $q^*, q' \in P$ such that $q \Rightarrow q^* \xrightarrow{u} q'$ and $R(p,q^*)$, $R(p',q')$.

ii. if $R(p,q)$ and $q \xrightarrow{u} q'$ ($u \in A_{\tau\sigma}$), then either:
   a. $u \equiv \tau$ and $R(p,q')$, or
   b. there are $p^*, p' \in P$ such that $p \Rightarrow p^* \xrightarrow{u} p'$ and $R(p^*,q)$, $R(p',q')$.

iii. if $R(p,q)$ and $p \xrightarrow{a} \surd$ ($a \in A_\tau$) then there is $q' \in P$ such that $q \Rightarrow q' \xrightarrow{a} \surd$ and $R(p,q')$.

iv. if $R(p,q)$ and $q \xrightarrow{a} \surd$ ($a \in A_\tau$) then there is $p' \in P$ such that $p \Rightarrow p' \xrightarrow{a} \surd$ and $R(p',q)$.

v. if $R(p,q)$ then $\mathrm{ID}(p)$ iff $\mathrm{ID}(q)$.

Two terms p,q are *branching tail bisimulation equivalent*, $p \leftrightarrow_{bt} q$, if there is a branching tail bisimulation relating p and q.

If R is a branching tail bisimulation, then we say that the related pair p,q satisfies the *root condition* if:

vi. if $p \xrightarrow{u} p'$ ($u \in A_{\tau\sigma}$), then there is q' such that $q \xrightarrow{u} q'$ and $R(p',q')$

vii. if $q \xrightarrow{u} q'$ ($u \in A_{\tau\sigma}$), then there is p' such that $p \xrightarrow{u} p'$ and $R(p',q')$
viii. $p \xrightarrow{a} \surd$ ($a \in A_\tau$) iff $q \xrightarrow{a} \surd$.
Two terms p,q are *rooted branching tail bisimulation equivalent*, $g \underline{\leftrightarrow}_{rbt}\, h$, if there is a branching bisimulation R relating p and q, such that the pair p,q satisfies the root condition, and all related pairs that can be reached from p and q by only performing σ-steps also satisfy the root condition.
Thus the root condition amounts to the condition that the bisimulation is strong as long as no action is executed.

We can prove that rooted branching tail bisimulation is a congruence relation, and thus we obtain the algebra $P/\underline{\leftrightarrow}_{rbt}$ of closed process expressions modulo rooted branching tail bisimulation. We can establish that the algebra $P/\underline{\leftrightarrow}_{rbt}$ satisfies all laws of $\mathrm{BPA}^{-}_{\mathrm{drt}}$.

Note that if we take the reduced model obtained by omitting the immediate deadlock constant, then we can define a notion of branching bisimulation without the tail condition, where a term can be related to $\surd$. Using the embedding of discrete time process algebra into real time process algebra given in [BAB92, 95] we find that our notion of silent step in time is in line with the notion of branching bisimulation in time of [KLU93] (albeit that he has no tail condition).

2.4 AXIOMATISATION FOR SILENT STEP.

Now we want to formulate algebraic laws for the silent step, that hold in the algebra $P/\underline{\leftrightarrow}_{rbt}$. We cannot just transpose the well-known laws, because of the special status of σ-steps. An example: $cts(a)\cdot(cts(\tau)\cdot(\sigma_{rel}(cts(b)) + \sigma_{rel}(cts(c))) + \sigma_{rel}(cts(b)))$ is not rooted branching tail bisimilation equivalent to $cts(a)\cdot(\sigma_{rel}(cts(b)) + \sigma_{rel}(cts(c)))$, as in the first term, the choice for b can be made by the execution of the σ-step, and in the second term, the choice must be made after the σ-step (it is equal to $cts(a)\cdot\sigma_{rel}(cts(b) + cts(c))$). In order to ensure that we do not split terms that both have an initial σ-step, we use the time out operator ν_{rel}.

In table 3, DRTB1 and DRTB2 are variants of the branching law B2 ($x\cdot(\tau\cdot(y + z) + y) = x\cdot(y + z)$). In DRTB1, we have the case where y does not have an initial time step, in DRTB2 this is the case for z. We add $cts(\delta)$ to ensure that the expression following $cts(\tau)$ does not equal $\dot{\delta}$. A simple instance is the identity $x\cdot cts(\tau)\cdot cts(\tau) = x\cdot cts(\tau)$. However, we do not have the law $x\cdot cts(\tau) = x$ (the counterpart of the branching law B1), as $cts(a)\cdot cts(\tau)\cdot\dot{\delta}$ must be distinguished from $cts(a)\cdot\dot{\delta}$ (this can be appreciated in a setting with parallel composition: the first term allows execution of actions in the current time slice from a parallel component after the execution of a, the second term does not).

The law DRTB3 allows to omit a τ-step following one or more time steps. A simple instance is $cts(a)\cdot\sigma_{rel}(cts(\tau)\cdot cts(b)) = cts(a)\cdot\sigma_{rel}(cts(b))$.

$x\cdot(cts(\tau)\cdot(\nu_{rel}(y) + z + cts(\delta)) + \nu_{rel}(y)) = x\cdot(\nu_{rel}(y) + z + cts(\delta))$ DRTB1
$x\cdot(cts(\tau)\cdot(y + \nu_{rel}(z) + cts(\delta)) + y) = x\cdot(y + \nu_{rel}(z) + cts(\delta))$ DRTB2
$cts(a)\cdot x = cts(a)\cdot y \Rightarrow cts(a)\cdot(\sigma_{rel}(x) + \nu_{rel}(z)) = cts(a)\cdot(\sigma_{rel}(y) + \nu_{rel}(z))$ DRTB3

TABLE 3. $\mathrm{BPA}^{-}_{\mathrm{drt}}\tau = \mathrm{BPA}^{-}_{\mathrm{drt}}$ + DCS + DRTB1-3 ($a \in A_\tau$).

2.5 PROPOSITION: The model $P/\underline{\leftrightarrow}_{rbt}$ satisfies the axioms DRTB1-3. We leave it as an open problem, whether the axiomatisation of $BPA_{d\bar{r}t}\tau$ is complete for the model $P/\underline{\leftrightarrow}_{rbt}$.

Using the interpretation of a,δ,τ by $cts(a)$, $cts(\delta)$, $cts(\tau)$, we do not obtain the time free theory $BPA_{\delta}{}^{\tau}$ of [GLW91, BAW90] as an SRM specification of $BPA_{d\bar{r}t}\tau$. If we reduce the initial algebra by omitting $\dot{\delta}$ and the σ_{rel} and ν_{rel} operators, then the first branching law $x{\cdot}\tau = x$ will not hold, but instead $x{\cdot}\tau{\cdot}y = x{\cdot}y$. We can nevertheless obtain $BPA_{\delta}{}^{\tau}$ as an SRM specifications, if we use a different interpretation of the constants: define $ctstau(a) = cts(a){\cdot}cts(\tau)$ $(a \in A_{\tau})$, then $BPA_{\delta}{}^{\tau}$ becomes an SRM specification of $BPA_{d\bar{r}t}\tau$ with the interpretation of a,τ,δ as $ctstau(a)$, $ctstau(\tau)$, $cts(\delta)$.

2.6 PROPOSITION. The following laws are derivable from $BPA_{d\bar{r}t}\tau$:

1. $ctstau(a){\cdot}ctstau(\tau) = ctstau(a)$
2. $x{\cdot}(ctstau(\tau){\cdot}(\nu_{rel}(y) + z + cts(\delta)) + \nu_{rel}(y)) = x{\cdot}(\nu_{rel}(y) + z + cts(\delta))$
3. $x{\cdot}(ctstau(\tau){\cdot}(y + \nu_{rel}(z) + cts(\delta)) + y) = x{\cdot}(y + \nu_{rel}(z) + cts(\delta))$

PROOF: 1 is straightforward; for 2, we calculate:
$x{\cdot}(ctstau(\tau){\cdot}(\nu_{rel}(y) + z + cts(\delta)) + \nu_{rel}(y)) = x{\cdot}(cts(\tau){\cdot}(cts(\tau){\cdot}(\nu_{rel}(y) + z + cts(\delta)) + cts(\delta)) +$
$+ \nu_{rel}(y)) = x{\cdot}(cts(\tau){\cdot}(\nu_{rel}(y) + z + cts(\delta)) + \nu_{rel}(y)) = x{\cdot}(y + \nu_{rel}(z) + cts(\delta))$. 3 goes similarly.

As a consequence, if we take the SRM obtained by omitting the $cts(a)$ constants for $a \in A_{\tau}$, then we can prove $x{\cdot}ctstau(\tau) = x$ for all closed terms. Thus, we have embedded the time free theory into the relative discrete time theory.

2.7 DELAYABLE ACTIONS.
However, this is not the embedding of the time free theory into the discrete time theory that we want: it reduces the whole world to one time slice. Rather, we want to interpret the time free actions as actions that can occur in any time slice. Following [BAB95], we extend $BPA_{d\bar{r}t}$ to BPA_{drt} by introducing constants $ats(a)$ (for $a \in A_{\tau\delta}$). $ats(a)$ executes a in an arbitrary time slice, followed by immediate termination. We define these constants using the operator $\lfloor\_\rfloor^{\omega}$, the *unbounded start delay* operator of [NIS94]: $\lfloor x\rfloor^{\omega}$ can start the execution of x in the current time slice, or delay unchanged to the next time slice. The defining axiom for this operator takes the form of a recursive equation. In order to prove identities for the unbounded start delay operator, we need a restricted form of the Recursive Specification Principle RSP of [BEK86]. We call this RSP(USD). We give operational rules for the new operator in table 5. Following [BAB95] we can prove that BPA_{drt} is a complete axiomatisation of the model of closed process expressions modulo strong bisimulation equivalence. We can also prove that the model $P/\underline{\leftrightarrow}_{rbt}$ satisfies the axioms of BPA_{drt}.

$\lfloor x\rfloor^{\omega} = \nu_{rel}(x) + \sigma_{rel}(\lfloor x\rfloor^{\omega})$	USD
$ats(a) = \lfloor cts(a)\rfloor^{\omega}$	ATS
$y = \nu_{rel}(x) + \sigma_{rel}(y) \Rightarrow y = \lfloor x\rfloor^{\omega}$	RSP(USD)

TABLE 4. $BPA_{drt} = BPA_{d\bar{r}t}$ + USD, ATS $(a \in A_{\tau\delta})$.

$$\frac{x \xrightarrow{a} x'}{\lfloor x \rfloor^{\omega} \xrightarrow{a} x'} \qquad \frac{x \xrightarrow{a} \surd}{\lfloor x \rfloor^{\omega} \xrightarrow{a} \surd} \qquad \lfloor x \rfloor^{\omega} \xrightarrow{\sigma} \lfloor x \rfloor^{\omega}$$

TABLE 5. Operational rules for unbounded start delay ($a \in A_\tau$).

2.8 PROPOSITION. The following identities are derivable from BPA_{drt} + RSP(USD):

1. $\lfloor \dot{\delta} \rfloor^{\omega} = ats(\delta)$	5. $ats(a) = cts(a) + \sigma_{rel}(ats(a))$	(ADRT)
2. $\lfloor x + y \rfloor^{\omega} = \lfloor x \rfloor^{\omega} + \lfloor y \rfloor^{\omega}$	6. $ats(a) + ats(\delta) = ats(a)$	(A6A)
3. $\lfloor x \cdot y \rfloor^{\omega} = \lfloor x \rfloor^{\omega} \cdot y$	7. $ats(\delta) \cdot x = ats(\delta)$	(A7)
4. $\lfloor \sigma_{rel}(x) \rfloor^{\omega} = ats(\delta)$.		

2.9 AXIOMATISATION FOR SILENT STEP.
We have one additional axiom for the constant $ats(\tau)$, DTB4.

$x \cdot (ats(\tau) \cdot \lfloor y + z \rfloor^{\omega} + \lfloor y \rfloor^{\omega}) = x \cdot \lfloor y + z \rfloor^{\omega}$	DTB4

TABLE 6. $BPA_{drt}\tau = BPA_{drt}$ + DTB4.

2.10 PROPOSITION: The model $P/\underline{\leftrightarrow}_{rbt}$ satisfies the axioms of $BPA_{drt}\tau$. Again, completeness is an open problem.

Again, we cannot obtain the time free theory $BPA_\delta{}^\tau$ as an SRM specification, interpreting a by $ats(a)$, since $ats(a) \cdot ats(\tau)$ is not branching tail bisimilar to $ats(a)$: the first term can still perform time steps after executing the action. Note that the second branching law is valid, as the unbounded start delay is the identity on all time free processes. In order to achieve an SRM specification, nonetheless, we will define a different interpretation of time free atoms in our timed setting. We define constants $atstau(a)$ as follows ($a \in A_\tau$): $atstau(a) = ats(a) \cdot ats(\tau)$.
Interpreting a, τ, δ by $atstau(a)$, $atstau(\tau)$, $ats(\delta)$, we do obtain $BPA_\delta{}^\tau$ as an SRM specification.

2.11 PROPOSITION. The following laws are derivable from $BPA_{drt}\tau$:
1. $atstau(a) \cdot atstau(\tau) = atstau(a)$
2. $x \cdot (atstau(\tau) \cdot \lfloor y + z \rfloor^{\omega} + \lfloor y \rfloor^{\omega}) = x \cdot \lfloor y + z \rfloor^{\omega}$

PROOF: as before.

2.12 ADDITIONAL OPERATOR.
We explore the theory $BPA_{drt}\tau$ a bit further by adding an extra operator: $\overline{v}_{rel}$ is the counterpart of the operator v_{rel}, it gives the part of the process that starts with an initial time step. $\overline{v}_{rel}$ was called initialisation in the following time slice in [BAB92]. $\overline{v}_{rel}(x)$ was notated $1 \gg_{dt} x$. The axiom DRTSA, the Discrete Relative Time Slice Axiom, allows to split each term into two components. Both models $P/\underline{\leftrightarrow}$, $P/\underline{\leftrightarrow}_{rbt}$ satisfy the axioms DRTSA, DNS1-4, if we add the operational rules in table 8.

$x = v_{rel}(x) + \bar{v}_{rel}(x)$	DRTSA		
$\bar{v}_{rel}(\dot{\delta}) = \dot{\delta}$	DNS1	$\bar{v}_{rel}(x + y) = \bar{v}_{rel}(x) + \bar{v}_{rel}(y)$	DNS3
$\bar{v}_{rel}(cts(a)) = cts(\delta)$	DNS2	$\bar{v}_{rel}(x \cdot y) = \bar{v}_{rel}(x) \cdot y$	DNS4

TABLE 7. Axioms for the additional operator ($a \in A_\tau$).

$$\frac{x \xrightarrow{\sigma} x'}{\bar{v}_{rel}(x) \xrightarrow{\sigma} x'} \qquad \frac{ID(x)}{ID(\bar{v}_{rel}(x))}$$

TABLE 8. Operational rules for the additional operator ($a \in A_\tau$).

2.13 PARALLEL COMPOSITION.

The extension of the theory $BPA_{drt}\tau$ with parallel composition, with or without communication, can be done along the lines of [BAB95]. The additional syntax has binary operators $\|$ (merge), $\mathbin{\|\!\!\!\_}$ (left merge), $|$ (communication merge), unary operators ∂_H (encapsulation operator, for $H \subseteq A$), τ_I (abstraction operator, for $I \subseteq A$). We present axioms for $ACP_{drt}\tau$ in table 9, operational rules in table 10. We assume given a partial commutative associative communication function γ: $A \times A \to A$.

$cts(a) \mid cts(b) = cts(c)$	if $\gamma(a,b)=c$	DRTCF1
$cts(a) \mid cts(b) = cts(\delta)$	otherwise	DRTCF2
$x \parallel y = x \mathbin{\|\!\!\!\_} y + y \mathbin{\|\!\!\!\_} x + x \mid y$		CM1
$\dot{\delta} \mathbin{\|\!\!\!\_} x = \dot{\delta}$		LMID1
$x \mathbin{\|\!\!\!\_} \dot{\delta} = \dot{\delta}$		LMID2
$(x + y) \mathbin{\|\!\!\!\_} z = x \mathbin{\|\!\!\!\_} z + y \mathbin{\|\!\!\!\_} z$		CM4
$cts(a) \mathbin{\|\!\!\!\_} (x + cts(\delta)) = cts(a) \cdot (x + cts(\delta))$		DRTCM2
$cts(a) \cdot x \mathbin{\|\!\!\!\_} (y + cts(\delta)) = cts(a) \cdot (x \parallel (y + cts(\delta)))$		DRTCM3
$\sigma_{rel}(x) \mathbin{\|\!\!\!\_} (\sigma_{rel}(y) + v_{rel}(z)) = \sigma_{rel}(x \mathbin{\|\!\!\!\_} y)$		DRT5

$\dot{\delta} \mid x = \dot{\delta}$	CMID1	$cts(a) \cdot x \mid cts(b) = cts(a \mid b) \cdot x$	DRTCM5
$x \mid \dot{\delta} = \dot{\delta}$	CMID2	$cts(a) \mid cts(b) \cdot x = cts(a \mid b) \cdot x$	DRTCM6
$(x+y) \mid z = x \mid z + y \mid z$	CM8	$cts(a) \cdot x \mid cts(b) \cdot y = cts(a \mid b) \cdot (x \parallel y)$	DRTCM7
$x \mid (y+z) = x \mid y + x \mid z$	CM9	$v_{rel}(x) \mid \sigma_{rel}(y) = cts(\delta)$	DRT6
$\sigma_{rel}(x) \mid \sigma_{rel}(y) = \sigma_{rel}(x \mid y)$	DRT8	$\sigma_{rel}(x) \mid v_{rel}(y) = cts(\delta)$	DRT7

$\partial_H(\dot{\delta}) = \dot{\delta}$	DID	$\partial_H(cts(a)) = cts(a)$	if $a \notin H$	DRTD1
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3	$\partial_H(cts(a)) = cts(\delta)$	if $a \in H$	DRTD2
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	D4	$\partial_H(\sigma_{rel}(x)) = \sigma_{rel}(\partial_H(x))$		DRT9
$\tau_I(\dot{\delta}) = \dot{\delta}$	TIID	$\tau_I(cts(a)) = cts(a)$	if $a \notin I$	DRTTI1
$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$	TI3	$\tau_I(cts(a)) = cts(\delta)$	if $a \in I$	DRTTI2
$\tau_I(x \cdot y) = \tau_I(x) \cdot \tau_I(y)$	TI4	$\tau_I(\sigma_{rel}(x)) = \sigma_{rel}(\tau_I(x))$		DRT10

TABLE 9. Additional axioms for $ACP_{drt}\tau$ ($a \in A_{\delta\tau}$).

$$\frac{x \xrightarrow{a} x',\ \neg\, \mathrm{ID}(y)}{x \| y \xrightarrow{a} x' \| y,\ y \| x \xrightarrow{a} y \| x',\ x \mathbin{\underline{\|}} y \xrightarrow{a} x' \| y} \qquad \frac{x \xrightarrow{a} \surd,\ \neg\, \mathrm{ID}(y)}{x \| y \xrightarrow{a} y,\ y \| x \xrightarrow{a} y,\ x \mathbin{\underline{\|}} y \xrightarrow{a} y}$$

$$\frac{x \xrightarrow{\sigma} x',\ y \xrightarrow{\sigma} y'}{x \| y \xrightarrow{\sigma} x' \| y',\ x \mathbin{\underline{\|}} y \xrightarrow{\sigma} x' \mathbin{\underline{\|}} y',\ x \mid y \xrightarrow{\sigma} x' \mid y'}$$

$$\frac{\mathrm{ID}(x)}{\mathrm{ID}(x \| y),\ \mathrm{ID}(y \| x),\ \mathrm{ID}(x \mathbin{\underline{\|}} y),\ \mathrm{ID}(y \mathbin{\underline{\|}} x),\ \mathrm{ID}(x \mid y),\ \mathrm{ID}(y \mid x)}$$

$$\frac{x \xrightarrow{a} x',\ y \xrightarrow{b} y',\ a \mid b = c}{x \| y \xrightarrow{c} x' \| y',\ x \mid y \xrightarrow{c} x' \| y'} \qquad \frac{x \xrightarrow{a} x',\ y \xrightarrow{b} \surd,\ a \mid b = c}{x \| y \xrightarrow{c} x',\ x \mid y \xrightarrow{c} x',\ y \| x \xrightarrow{c} x',\ y \mid x \xrightarrow{c} x'}$$

$$\frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} \surd,\ a \mid b = c}{x \| y \xrightarrow{c} \surd,\ x \mid y \xrightarrow{c} \surd}$$

$$\frac{x \xrightarrow{a} x',\ a \notin H}{\partial_H(x) \xrightarrow{a} \partial_H(x')} \qquad \frac{x \xrightarrow{\sigma} x'}{\partial_H(x) \xrightarrow{\sigma} \partial_H(x')} \qquad \frac{x \xrightarrow{a} \surd,\ a \notin H}{\partial_H(x) \xrightarrow{a} \surd} \qquad \frac{\mathrm{ID}(x)}{\mathrm{ID}(\partial_H(x))}$$

$$\frac{x \xrightarrow{a} x',\ a \notin I}{\tau_I(x) \xrightarrow{a} \tau_I(x')} \qquad \frac{x \xrightarrow{a} x',\ a \in I}{\tau_I(x) \xrightarrow{\tau} \tau_I(x')} \qquad \frac{x \xrightarrow{a} \surd,\ a \notin I}{\tau_I(x) \xrightarrow{a} \surd} \qquad \frac{x \xrightarrow{a} \surd,\ a \in I}{\tau_I(x) \xrightarrow{\tau} \surd}$$

$$\frac{x \xrightarrow{\sigma} x'}{\tau_I(x) \xrightarrow{\sigma} \tau_I(x')} \qquad \frac{\mathrm{ID}(x)}{\mathrm{ID}(\tau_I(x))}$$

TABLE 10. Additional operational rules for $\mathrm{ACP}_{\mathrm{drt}}\tau$.

2.14 PROPOSITION. We derive the following identities for time free atoms in $\mathrm{ACP}_{\mathrm{drt}}\tau$ + RSP(USD):

1. $\mathrm{ats}(a) \mathbin{\underline{\|}} \lfloor x \rfloor^{\omega} = \mathrm{ats}(a) \cdot \lfloor x \rfloor^{\omega}$ (CM2)	6. $\partial_H(\mathrm{ats}(a)) = \mathrm{ats}(a)$	if $a \notin H$	(D1)
2. $\mathrm{ats}(a) \cdot x \mathbin{\underline{\|}} \lfloor y \rfloor^{\omega} = \mathrm{ats}(a) \cdot (x \| \lfloor y \rfloor^{\omega})$ (CM3)	7. $\partial_H(\mathrm{ats}(a)) = \mathrm{ats}(\delta)$	if $a \in H$	(D2)
3. $\mathrm{ats}(a) \cdot x \mid \mathrm{ats}(b) = \mathrm{ats}(a \mid b) \cdot x$ (CM5)	8. $\tau_I(\mathrm{ats}(a)) = \mathrm{ats}(a)$	if $a \notin I$	(TI1)
4. $\mathrm{ats}(a) \mid \mathrm{ats}(b) \cdot x = \mathrm{ats}(a \mid b) \cdot x$ (CM6)	9. $\tau_I(\mathrm{ats}(a)) = \mathrm{ats}(\tau)$	if $a \in I$	(TI2)
5. $\mathrm{ats}(a) \cdot x \mid \mathrm{ats}(b) \cdot y = \mathrm{ats}(a \mid b) \cdot (x \| y)$ (CM7)			

The following identity, useful in verifications, can be proved for all closed terms x, y:

10. $\mathrm{cts}(a) \cdot (\mathrm{cts}(\tau) \cdot (x + \mathrm{cts}(\delta)) \| y) = \mathrm{cts}(a) \cdot ((x + \mathrm{cts}(\delta)) \| y)$.

3. DISCRETE TIME PROCESS ALGEBRA WITH ABSOLUTE TIMING.

We present a version of the theory in section 2 using absolute timing, where all timing is related to a global clock.

3.1 BASIC PROCESS ALGEBRA.

We start with constants $\mathrm{fts}(a)$, denoting a in the first time slice ($a \in A_{\tau\delta}$), followed by immediate termination. Besides, we have operators $+, \cdot$ as before. In addition, we have the absolute discrete time unit delay σ_{abs}. Axiom DAT4 uses the absolute value operator. This operator turns out to be the identity for all processes using absolute timing only: it initialises a process in the first time slice (the numbering of the axioms is taken from [BAB95]). Note that in the term $\sigma_{\mathrm{abs}}(\mathrm{fts}(a))\cdot\mathrm{fts}(\delta)$, after execution of the a in slice 2, an immediate deadlock will occur. This term is different from $\sigma_{\mathrm{abs}}(\mathrm{fts}(a)\cdot\mathrm{fts}(\delta))$, where after the execution of a in slice 2, further activity in this slice can take place (of a parallel process). We conclude that in the absolute time theory, the immediate deadlock constant $\dot{\delta}$ is necessary. The axiomatisation of $\mathrm{BPA}_{\bar{\mathrm{d}}\mathrm{at}}$ + DFS adds the axioms of table 11 to the axioms A1-5, A6ID, A7ID (see the left hand side of table 1).

$\sigma_{\mathrm{abs}}(\dot{\delta}) = \mathrm{fts}(\delta)$	DAT1	$\lvert\dot{\delta}\rvert = \dot{\delta}$	AV1
$\mathrm{fts}(a) + \mathrm{fts}(\delta) = \mathrm{fts}(a)$	DAT2	$\lvert x + y\rvert = \lvert x\rvert + \lvert y\rvert$	AV3
$\sigma_{\mathrm{abs}}(x) + \sigma_{\mathrm{abs}}(y) = \sigma_{\mathrm{abs}}(x + y)$	DAT3	$\lvert x\cdot y\rvert = \lvert x\rvert\cdot y$	AV4
$\sigma_{\mathrm{abs}}(x)\cdot\dot{\delta} = \sigma_{\mathrm{abs}}(x\cdot\dot{\delta})$	DAT4	$\lvert\mathrm{fts}(a)\rvert = \mathrm{fts}(a)$	AV8
$\sigma_{\mathrm{abs}}(x)\cdot(\mathrm{fts}(a) + y) = \sigma_{\mathrm{abs}}(x)\cdot y$	DAT5	$\lvert\sigma_{\mathrm{abs}}(x)\rvert = \sigma_{\mathrm{abs}}(\lvert x\rvert)$	AV9
$\sigma_{\mathrm{abs}}(x)\cdot(\mathrm{fts}(a)\cdot y + z) = \sigma_{\mathrm{abs}}(x)\cdot z$	DAT6	$\lvert\nu_{\mathrm{abs}}(x)\rvert = \nu_{\mathrm{abs}}(\lvert x\rvert)$	AV10
$\sigma_{\mathrm{abs}}(x)\cdot\sigma_{\mathrm{abs}}(y) = \sigma_{\mathrm{abs}}(x\cdot\lvert y\rvert)$	DAT7		
$\nu_{\mathrm{abs}}(\dot{\delta}) = \dot{\delta}$	DFS1	$\nu_{\mathrm{abs}}(x\cdot y) = \nu_{\mathrm{abs}}(x)\cdot y$	DFS4
$\nu_{\mathrm{abs}}(\mathrm{fts}(a)) = \mathrm{fts}(a)$	DFS2	$\nu_{\mathrm{abs}}(x + y) = \nu_{\mathrm{abs}}(x) + \nu_{\mathrm{abs}}(y)$	DFS5
$\nu_{\mathrm{abs}}(\sigma_{\mathrm{abs}}(x)) = \mathrm{fts}(\delta)$	DFS3		

TABLE 11. $\mathrm{BPA}_{\bar{\mathrm{d}}\mathrm{at}}$ + DFS ($a \in A_{\delta\tau}$).

3.2 STRUCTURED OPERATIONAL SEMANTICS.

The operational rules are more complicated in this case, as we have to keep track of which time slice we are in, we have to keep track of the global clock. $\langle x,n\rangle$ denotes x in the $(n+1)$st time slice. We have:

- if $\langle x,n\rangle \xrightarrow{\sigma} \langle x',n'\rangle$, then $x \equiv x'$, $n' = n+1$
- if $\langle x,n\rangle \xrightarrow{a} \langle x',n'\rangle$ or $\langle x,n\rangle \xrightarrow{a} \langle\surd,n'\rangle$, then $n' = n$.

The operational rules for the absolute value operator are trivial.

$$\langle \mathrm{fts}(a), 0\rangle \xrightarrow{a} \langle \surd, 0\rangle \qquad \mathrm{ID}(\langle \dot{\delta}, n\rangle) \qquad \mathrm{ID}(\langle \mathrm{fts}(a), n+1\rangle)$$

$$\frac{\neg\, \mathrm{ID}(\langle x,0\rangle)}{\langle \sigma_{abs}(x),0\rangle \xrightarrow{\sigma} \langle \sigma_{abs}(x),1\rangle} \qquad \frac{\mathrm{ID}(\langle x,n\rangle)}{\mathrm{ID}(\langle \sigma_{abs}(x),n+1\rangle)} \qquad \frac{\langle x,n\rangle \xrightarrow{\sigma} \langle x,n+1\rangle}{\langle \sigma_{abs}(x),n+1\rangle \xrightarrow{\sigma} \langle \sigma_{abs}(x),n+2\rangle}$$

$$\frac{\langle x,n\rangle \xrightarrow{a} \langle x',n\rangle}{\langle \sigma_{abs}(x),n+1\rangle \xrightarrow{a} \langle \sigma_{abs}(x'),n+1\rangle} \qquad \frac{\langle x,n\rangle \xrightarrow{a} \langle \surd,n\rangle}{\langle \sigma_{abs}(x),n+1\rangle \xrightarrow{a} \langle \surd,n+1\rangle}$$

$$\frac{\langle x,n\rangle \xrightarrow{a} \langle x',n\rangle}{\langle x+y,n\rangle \xrightarrow{a} \langle x',n\rangle,\ \langle y+x,n\rangle \xrightarrow{a} \langle x',n\rangle} \qquad \frac{\langle x,n\rangle \xrightarrow{a} \langle \surd,n\rangle}{\langle x+y,n\rangle \xrightarrow{a} \langle \surd,n\rangle,\ \langle y+x,n\rangle \xrightarrow{a} \langle \surd,n\rangle}$$

$$\frac{\langle x,n\rangle \xrightarrow{\sigma} \langle x,n+1\rangle}{\langle x+y,n\rangle \xrightarrow{\sigma} \langle x+y,n+1\rangle,\ \langle y+x,n\rangle \xrightarrow{\sigma} \langle y+x,n+1\rangle} \qquad \frac{\mathrm{ID}(\langle x,n\rangle),\ \mathrm{ID}(\langle y,n\rangle)}{\mathrm{ID}(\langle x+y,n\rangle)}$$

$$\frac{\langle x,n\rangle \xrightarrow{a} \langle x',n\rangle}{\langle x\cdot y,n\rangle \xrightarrow{a} \langle x'\cdot y,n\rangle} \qquad \frac{\langle x,n\rangle \xrightarrow{\sigma} \langle x,n+1\rangle}{\langle x\cdot y,n\rangle \xrightarrow{\sigma} \langle x\cdot y,n+1\rangle} \qquad \frac{\langle x,n\rangle \xrightarrow{a} \langle \surd,n\rangle}{\langle x\cdot y,n\rangle \xrightarrow{a} \langle y,n\rangle} \qquad \frac{\mathrm{ID}(\langle x,n\rangle)}{\mathrm{ID}(\langle x\cdot y,n\rangle)}$$

$$\frac{\langle x,0\rangle \xrightarrow{a} \langle x',0\rangle}{\langle \nu_{abs}(x),0\rangle \xrightarrow{a} \langle x',0\rangle} \qquad \frac{\langle x,0\rangle \xrightarrow{a} \langle \surd,0\rangle}{\langle \nu_{abs}(x),0\rangle \xrightarrow{a} \langle \surd,0\rangle} \qquad \frac{\mathrm{ID}(\langle x,0\rangle)}{\mathrm{ID}(\langle \nu_{abs}(x),0\rangle)} \qquad \mathrm{ID}(\langle \nu_{abs}(x),n+1\rangle)$$

TABLE 12. Operational semantics of $\mathrm{BPA}_{\mathrm{d\bar{a}t}}$ + DFS ($a \in A_\tau$).

3.3 BISIMULATION.

We also have to adapt the definition of bisimulation. A *strong bisimulation* is a binary relation R on P × ℕ such that ($u \in A_{\tau\sigma}$):

i. whenever $R(\langle s, n\rangle, \langle s', n'\rangle)$ then $n = n'$ and $\mathrm{ID}(\langle s, n\rangle)$ iff $\mathrm{ID}(\langle s', n'\rangle)$.

ii. whenever $R(\langle s, n\rangle, \langle t, n\rangle)$ and $\langle s,n\rangle \xrightarrow{u} \langle s',n'\rangle$, then there is a term t' such that $\langle t,n\rangle \xrightarrow{u} \langle t',n'\rangle$ and $R(\langle s', n'\rangle, \langle t', n'\rangle)$

iii. whenever $R(\langle s, n\rangle, \langle t, n\rangle)$ and $\langle t,n\rangle \xrightarrow{u} \langle t',n'\rangle$, then there is a term s' such that $\langle s,n\rangle \xrightarrow{u} \langle s',n'\rangle$ and $R(\langle s', n'\rangle, \langle t', n'\rangle)$

iv. whenever $R(\langle s, n\rangle, \langle t, n\rangle)$, then $\langle s,n\rangle \xrightarrow{a} \langle \surd,n\rangle$ iff $\langle t,n\rangle \xrightarrow{a} \langle \surd,n\rangle$ ($a \in A_\tau$).

We say process expressions x and y are *strong bisimulation equivalent*, denoted $x \leftrightarrow y$, if there exists a strong bisimulation with $R(\langle x,0\rangle, \langle y,0\rangle)$. From [BAB95] we know that the axiomatisation in table 9 is complete for the model of closed process expressions modulo strong bisimulation.

A *branching tail bisimulation* is a binary relation R on P × ℕ such that:

i. whenever $R(\langle s, n\rangle, \langle s', n'\rangle)$ then $n = n'$ and $\mathrm{ID}(\langle s, n\rangle)$ iff $\mathrm{ID}(\langle s', n'\rangle)$.

ii. whenever $R(\langle s, n\rangle, \langle t, n\rangle)$ and $\langle s,n\rangle \xrightarrow{u} \langle s',n'\rangle$ ($u \in A_{\tau\sigma}$), then either:

a. $u \equiv \tau$ and $R(\langle s', n\rangle, \langle t, n\rangle)$, or

b. there are $t^*, t' \in P$ such that $\langle t,n\rangle \Rightarrow \langle t^*,n\rangle \xrightarrow{u} \langle t',n'\rangle$ and $R(\langle s, n\rangle, \langle t^*, n\rangle)$, $R(\langle s', n'\rangle, \langle t', n'\rangle)$

iii. whenever $R(\langle s, n\rangle, \langle t, n\rangle)$ and $\langle t,n\rangle \xrightarrow{u} \langle t',n'\rangle$ ($u \in A_{\tau\sigma}$), then either:

a. $u \equiv \tau$ and $R(\langle s, n\rangle, \langle t', n\rangle)$, or

b. there are $s^*, s' \in P$ such that $\langle s,n\rangle \Rightarrow \langle s^*,n\rangle \xrightarrow{u} \langle s',n'\rangle$ and $R(\langle s^*, n\rangle, \langle t, n\rangle)$, $R(\langle s', n'\rangle, \langle t', n'\rangle)$

iv. if $R(\langle s, n\rangle, \langle t, n\rangle)$ and $\langle s,n\rangle \xrightarrow{a} \langle \surd,n\rangle$ ($a \in A_\tau$) then there is $t' \in P$ such that $\langle t,n\rangle \Rightarrow \langle t',n\rangle \xrightarrow{a} \langle \surd,n\rangle$ and $R(\langle s, n\rangle, \langle t', n\rangle)$.

v. if $R(\langle s, n\rangle, \langle t, n\rangle)$ and $\langle t,n\rangle \xrightarrow{a} \langle \surd,n\rangle$ ($a \in A_\tau$) then there is $s' \in P$ such that $\langle s,n\rangle \Rightarrow \langle s',n\rangle \xrightarrow{a} \langle \surd,n\rangle$ and $R(\langle s', n\rangle, \langle t, n\rangle)$.

We say process expressions x and y are *branching tail bisimilation equivalent*, denoted $x \underline{\leftrightarrow}_{bt}\, y$, if there exists a branching tail bisimulation with $R(\langle x,0\rangle, \langle y,0\rangle)$. Process expressions x and y are *rooted branching tail bisimilation equivalent*, denoted $x \underline{\leftrightarrow}_{rbt}\, y$, if there exists a branching tail bisimulation with $R(\langle x,0\rangle,\langle y,0\rangle)$, that is strong for all pairs that can be reached from $\langle x,0\rangle,\langle y,0\rangle$ by just performing time steps.

3.4 AXIOMATISATION FOR SILENT STEP.

Axioms for the silent step are comparable to the ones for relative time. The model of closed process expressions modulo rooted branching tail bisimulation satisfies the following axioms. Again, we can find the time free theory $BPA_\delta{}^\tau$ as an SRM specification by defining the constants $ftstau(a) = fts(a) \cdot fts(\tau)$.

$fts(a) \cdot (fts(\tau) \cdot (v_{abs}(x) + y + fts(\delta)) + v_{abs}(x)) = fts(a) \cdot (v_{abs}(x) + y + fts(\delta))$	DATB1
$fts(a) \cdot (fts(\tau) \cdot (x + v_{abs}(y) + fts(\delta)) + x) = fts(a) \cdot (x + v_{abs}(y) + fts(\delta))$	DATB2
$fts(a) \cdot x = fts(a) \cdot y \Rightarrow fts(a) \cdot (\sigma_{abs}(x) + v_{abs}(z)) = fts(a) \cdot (\sigma_{abs}(y) + v_{abs}(z))$	DATB3

TABLE 13. Axioms for $BPA_{\bar{d}at}\tau$.

3.5 DELAYABLE ACTIONS.

The extension with delayable actions is not so smooth as in the relative time case. We can define a recursive equation for the unbounded start delay operator, but it uses the operator μ that we will only define in section 4. Instead, we define the unbounded start delay operator by structural induction. Also, time free atoms cannot simply be defined using the unbounded start delay operator. Instead, we will define them by a recursive equation. The theory BPA_{dat} adds the axioms in table 14 to $BPA_{\bar{d}at}$ + DFS. Operational rules are presented in table 15.

$\lfloor \dot{\delta} \rfloor^\omega = ats(\delta)$	USD1	$ats(a) = fts(a) + \sigma_{abs}(ats(a))$	ADAT
$\lfloor fts(a) \rfloor^\omega = fts(a) + ats(\delta)$	USD2	$\lvert ats(a) \rvert = ats(a)$	AV2
$\lfloor \sigma_{abs}(x) \rfloor^\omega = \sigma_{abs}(\lfloor x \rfloor^\omega)$	USD3	$v_{abs}(\lfloor x \rfloor^\omega) = v_{abs}(x)$	DFS6
$\lfloor x + y \rfloor^\omega = \lfloor x \rfloor^\omega + \lfloor y \rfloor^\omega$	USD4	$\lvert \lfloor x \rfloor^\omega \rvert = \lfloor x \rfloor^\omega$	AV11
$\lfloor x \cdot y \rfloor^\omega = \lfloor x \rfloor^\omega \cdot y$	USD5		

TABLE 14. Axioms for BPA_{dat}.

$$\langle \mathrm{ats}(a), n\rangle \xrightarrow{a} \langle \surd, n\rangle \qquad \langle \mathrm{ats}(a), n\rangle \xrightarrow{\sigma} \langle \mathrm{ats}(a), n+1\rangle \qquad \langle \mathrm{ats}(\delta), n\rangle \xrightarrow{\sigma} \langle \mathrm{ats}(\delta), n+1\rangle$$

$$\langle \lfloor x\rfloor^{\omega}, n\rangle \xrightarrow{\sigma} \langle \lfloor x\rfloor^{\omega}, n+1\rangle \qquad \frac{\langle x,n\rangle \xrightarrow{a} \langle x',n\rangle}{\langle \lfloor x\rfloor^{\omega},n\rangle \xrightarrow{a} \langle x',n\rangle} \qquad \frac{\langle x,n\rangle \xrightarrow{a} \langle \surd,n\rangle}{\langle \lfloor x\rfloor^{\omega},n\rangle \xrightarrow{a} \langle \surd,n\rangle}$$

TABLE 15. Operational rules for time free theory ($a \in A_\tau$).

3.6 AXIOMATISATION FOR SILENT STEP.

The axiomatisation for silent step has the same rule, DTB4, as in the relative time case. Again, we can find $BPA_\delta{}^\tau$ as an SRM specification by using the interpretation of a as atstau(a).

The extension with parallel composition can be found along the same lines as for the relative time case (see [BAB95]).

4. PARAMETRIC TIME.

In this section we integrate the absolute time and the relative time approach. All axioms presented in the previous sections are still valid for all parametric time processes. We obtain a finite axiomatization, that allows an elimination theorem. As a consequence, we can expand expressions like $\mathrm{cts}(a) \parallel \mathrm{fts}(b)$, $\mathrm{cts}(a) \parallel (\mathrm{fts}(b) + \mathrm{ats}(\delta))$.

In [BAB95], we introduced the operators $\odot$, the (relative) time spectrum combinator, and μ, the spectrum tail operator. The absolute value operator introduced in 3.1 can also be called the spectrum head operator. $P \odot Q$ is a process that when initialised in the first time slice behaves as $|P|$; when initialised in slice $n+1$ its behaviour is determined by Q as follows: initialise in slice n thereafter apply σ_{abs}. $\mu(X)$ computes a process such that $X = |X| \odot \mu(X)$. For a parametric discrete time process we have the time spectrum sequence $|X|, |\mu(X)|, |\mu^2(X)|, \ldots$. For each infinite sequence $(P_n)_{n \in \mathbb{N}}$ one may imagine a process P with $|\mu^n(P)| = P_n$ though not all such P can be finitely expressed.

4.1 BASIC PROCESS ALGEBRA.

$\lvert\mathrm{cts}(a)\rvert = \mathrm{fts}(a)$	AV10	$\mu(\dot\delta) = \dot\delta$	ST1
$\lvert\sigma_{rel}(X)\rvert = \sigma_{abs}(\lvert\mu(X)\rvert)$	AV11	$\mu(\mathrm{cts}(a)) = \mathrm{cts}(a)$	ST2
$\lvert\nu_{rel}(X)\rvert = \nu_{abs}(\lvert X\rvert)$	AV12	$\mu(X + Y) = \mu(X) + \mu(Y)$	ST3
$\sigma_{abs}(X) = \sigma_{abs}(\lvert X\rvert)$	AV13	$\mu(X \cdot Y) = \mu(X)\cdot\mu(Y)$	ST4
$\nu_{abs}(X) = \nu_{abs}(\lvert X\rvert)$	AV14	$\mu(\sigma_{rel}(X) = \sigma_{rel}(\mu(X))$	ST5
		$\mu(\nu_{rel}(x)) = \nu_{rel}(\mu(x))$	ST6
$\lvert X \odot Y\rvert = \lvert X\rvert$	SC1	$\mu(\mathrm{fts}(a)) = \dot\delta$	ST7
$\mu(X \odot Y) = Y$	SC2	$\mu(\sigma_{abs}(X)) = \lvert X\rvert$	ST8
$X = \lvert X\rvert \odot \mu(X)$	SC3	$\mu(\nu_{abs}(x)) = \dot\delta$	ST9
$\lfloor x\rfloor^{\omega} = \nu_{abs}(x) + \sigma_{abs}(\lfloor \mu(x)\rfloor^{\omega})$	USDA	$\mu(\lfloor x\rfloor^{\omega}) = \lfloor \mu(x)\rfloor^{\omega}$	ST10

TABLE 17. $BPA_{dpt} = BPA_{drt} + BPA_{dat}$ + AV10-14, ST1-10, SC1-3, USDA.

Note that the three axioms DAT5-7 can be replaced by $\sigma_{abs}(X) \cdot Y = \sigma_{abs}(X \cdot \mu(Y))$. We can define:

- x is an absolute time process iff $BPA_{dpt} \vdash x = |x|$
- x is a relative time process iff $BPA_{dpt} \vdash x = \mu(x)$.

5.2 BASIC FORM.

We claim that each BPA_{dpt} process expression can be written in the form

$$P_1 \odot P_2 \odot \ldots P_n \odot Q$$

(we omit brackets, using the convention that $\odot$ associates to the right), such that each P_i is a BPA_{dat}-term and Q is a BPA_{drt}-term.

The way we achieve this is by writing

$$X = |X| \odot |\mu(X)| \odot \ldots |\mu^n(X)| \odot \mu^{n+1}(X).$$

Now one can reduce each $|\mu^n(X)|$ to a BPA_{dat}-term and if n is sufficiently large, we can write $\mu^{n+1}(X))$ without any σ_{abs} or $fts(a)$ using ST1-7, so will be in the relative time signature.

We call $(|X|, |\mu(X)|, |\mu^2(X)|, \ldots)$ the time spectrum expansion sequence (TSS) of X.

Note that we obtain the following spectrum expansion for the v_{rel} operator:

$$v_{rel}(x) = v_{abs}(|x|) \odot v_{rel}(\mu(x)).$$

For further details, we refer to [BAB95].

REFERENCES.

[BAB91] J.C.M. BAETEN & J.A. BERGSTRA, *Real time process algebra,* Formal Aspects of Computing 3 (2), 1991, pp. 142-188.

[BAB92] J.C.M. BAETEN & J.A. BERGSTRA, *Discrete time process algebra (extended abstract),* in: Proc. CONCUR'92, Stony Brook (W.R. Cleaveland, ed.), LNCS 630, Springer 1992, pp. 401-420.

[BAB93] J.C.M. BAETEN & J.A. BERGSTRA, *Real space process algebra,* Formal Aspects of Computing 5 (6), 1993, pp. 481-529.

[BAB94] J.C.M. BAETEN & J.A. BERGSTRA, *On sequential composition, action prefixes and process prefix,* Formal Aspects of Computing 6 (3), 1994, pp. 250-268.

[BAB95] J.C.M. BAETEN & J.A. BERGSTRA, *Discrete time process algebra,* report P9208c, Programming Research Group, University of Amsterdam 1995. To appear in Formal Aspects of Computing.

[BAW90] J.C.M. BAETEN & W.P. WEIJLAND, *Process algebra,* Cambridge Tracts in Theor. Comp. Sci. 18, Cambridge University Press 1990.

[BEK85] J.A. BERGSTRA & J.W. KLOP, *Algebra of communicating processes with abstraction,* TCS 37 (1), 1985, pp. 77-121.

[BEK86] J.A. BERGSTRA & J.W. KLOP, *Verification of an alternating bit protocol by means of process algebra,* in: Math. Methods of Spec. and Synthesis of Software Systems '85 (W. Bibel & K.P. Jantke, eds.), Springer LNCS 215, 1986, pp. 9-23.

[BEK94] J.A. BERGSTRA & P. KLINT, *The toolbus - a component interconnection architecture -*, report P9408, Programming Research Group, University of Amsterdam 1994.

[BEK95] J.A. BERGSTRA & P. KLINT, *The discrete time toolbus*, report P9502, Programming Research Group, University of Amsterdam 1995.

[BEM95] J.A. BERGSTRA & C.A. MIDDELBURG, *A process algebra semantics for ϕSDL*, report LGPS 129, Dept. of Philosophy, Utrecht University 1995.

[BRHR84] S.D. BROOKES, C.A.R. HOARE & A.W. ROSCOE, *A theory of communicating sequential processes*, Journal of the ACM 31 (3), 1984, pp. 560-599.

[GLW91] R.J. VAN GLABBEEK & W.P. WEIJLAND, *Branching time and abstraction in bisimulation semantics*, CWI report CS-R9120, 1991.

[GRO90a] J.F. GROOTE, *Specification and verification of real time systems in ACP*, in: Proc. 10th Symp. on Protocol Specification, Testing and Verification, Ottawa (L. Logrippo, R.L. Probert & H. Ural, eds.), North-Holland, Amsterdam 1990, pp. 261-274.

[HER90] M. HENNESSY & T. REGAN, *A temporal process algebra*, report 2/90, University of Sussex 1990.

[KLU93] A.S. KLUSENER, *Models and axioms for a fragment of real time process algebra*, Ph.D. Thesis, Eindhoven University of Technology 1993.

[MIL89] R. MILNER, *Communication and concurrency*, Prentice-Hall 1989.

[MOT89] F. MOLLER & C. TOFTS, *A temporal calculus of communicating systems*, report LFCS-89-104, University of Edinburgh 1989.

[MOT90] F. MOLLER & C. TOFTS, *A temporal calculus of communicating systems*, in: Proc. CONCUR'90, Amsterdam (J.C.M. Baeten & J.W. Klop, eds.), LNCS 458, Springer 1990, pp. 401-415.

[NIS94] X. NICOLLIN & J. SIFAKIS, *The algebra of timed processes ATP: theory and application*, Information & Computation 114, 1994, pp. 131-178.

[NSVR90] X. NICOLLIN, J.-L. RICHIER, J. SIFAKIS & J. VOIRON, *ATP: an algebra for timed processes*, in Proc. IFIP TC2 Conf. on Progr. Concepts & Methods, Sea of Gallilee, Israel 1990.

[RER88] G.M. REED & A.W. ROSCOE, *A timed model for communicating sequential processes*, TCS 58, 1988, pp. 249-261.

[VER94] C. VERHOEF, *A congruence theorem for structured operational semantics with predicates and negative premises*, in: Proc. CONCUR'94, Uppsala (B. Jonsson & J. Parrow, eds.), Springer LNCS 836, 1994, pp. 433-448.

Computing Science Reports **Department of Mathematics and Computing Science**
Eindhoven University of Technology

In this series appeared:

93/01	R. van Geldrop	Deriving the Aho-Corasick algorithms: a case study into the synergy of programming methods, p. 36.
93/02	T. Verhoeff	A continuous version of the Prisoner's Dilemma, p. 17
93/03	T. Verhoeff	Quicksort for linked lists, p. 8.
93/04	E.H.L. Aarts J.H.M. Korst P.J. Zwietering	Deterministic and randomized local search, p. 78.
93/05	J.C.M. Baeten C. Verhoef	A congruence theorem for structured operational semantics with predicates, p. 18.
93/06	J.P. Veltkamp	On the unavoidability of metastable behaviour, p. 29
93/07	P.D. Moerland	Exercises in Multiprogramming, p. 97
93/08	J. Verhoosel	A Formal Deterministic Scheduling Model for Hard Real-Time Executions in DEDOS, p. 32.
93/09	K.M. van Hee	Systems Engineering: a Formal Approach Part I: System Concepts, p. 72.
93/10	K.M. van Hee	Systems Engineering: a Formal Approach Part II: Frameworks, p. 44.
93/11	K.M. van Hee	Systems Engineering: a Formal Approach Part III: Modeling Methods, p. 101.
93/12	K.M. van Hee	Systems Engineering: a Formal Approach Part IV: Analysis Methods, p. 63.
93/13	K.M. van Hee	Systems Engineering: a Formal Approach Part V: Specification Language, p. 89.
93/14	J.C.M. Baeten J.A. Bergstra	On Sequential Composition, Action Prefixes and Process Prefix, p. 21.
93/15	J.C.M. Baeten J.A. Bergstra R.N. Bol	A Real-Time Process Logic, p. 31.
93/16	H. Schepers J. Hooman	A Trace-Based Compositional Proof Theory for Fault Tolerant Distributed Systems, p. 27
93/17	D. Alstein P. van der Stok	Hard Real-Time Reliable Multicast in the DEDOS system, p. 19.
93/18	C. Verhoef	A congruence theorem for structured operational semantics with predicates and negative premises, p. 22.
93/19	G-J. Houben	The Design of an Online Help Facility for ExSpect, p.21.
93/20	F.S. de Boer	A Process Algebra of Concurrent Constraint Programming, p. 15.
93/21	M. Codish D. Dams G. Filé M. Bruynooghe	Freeness Analysis for Logic Programs - And Correctness, p. 24
93/22	E. Poll	A Typechecker for Bijective Pure Type Systems, p. 28.
93/23	E. de Kogel	Relational Algebra and Equational Proofs, p. 23.
93/24	E. Poll and Paula Severi	Pure Type Systems with Definitions, p. 38.
93/25	H. Schepers and R. Gerth	A Compositional Proof Theory for Fault Tolerant Real-Time Distributed Systems, p. 31.
93/26	W.M.P. van der Aalst	Multi-dimensional Petri nets, p. 25.
93/27	T. Kloks and D. Kratsch	Finding all minimal separators of a graph, p. 11.
93/28	F. Kamareddine and R. Nederpelt	A Semantics for a fine λ-calculus with de Bruijn indices, p. 49.
93/29	R. Post and P. De Bra	GOLD, a Graph Oriented Language for Databases, p. 42.
93/30	J. Deogun T. Kloks D. Kratsch H. Müller	On Vertex Ranking for Permutation and Other Graphs, p. 11.

16

93/31	W. Körver	Derivation of delay insensitive and speed independent CMOS circuits, using directed commands and production rule sets, p. 40.
93/32	H. ten Eikelder and H. van Geldrop	On the Correctness of some Algorithms to generate Finite Automata for Regular Expressions, p. 17.
93/33	L. Loyens and J. Moonen	ILIAS, a sequential language for parallel matrix computations, p. 20.
93/34	J.C.M. Baeten and J.A. Bergstra	Real Time Process Algebra with Infinitesimals, p.39.
93/35	W. Ferrer and P. Severi	Abstract Reduction and Topology, p. 28.
93/36	J.C.M. Baeten and J.A. Bergstra	Non Interleaving Process Algebra, p. 17.
93/37	J. Brunekreef J-P. Katoen R. Koymans S. Mauw	Design and Analysis of Dynamic Leader Election Protocols in Broadcast Networks, p. 73.
93/38	C. Verhoef	A general conservative extension theorem in process algebra, p. 17.
93/39	W.P.M. Nuijten E.H.L. Aarts D.A.A. van Erp Taalman Kip K.M. van Hee	Job Shop Scheduling by Constraint Satisfaction, p. 22.
93/40	P.D.V. van der Stok M.M.M.P.J. Claessen D. Alstein	A Hierarchical Membership Protocol for Synchronous Distributed Systems, p. 43.
93/41	A. Bijlsma	Temporal operators viewed as predicate transformers, p. 11.
93/42	P.M.P. Rambags	Automatic Verification of Regular Protocols in P/T Nets, p. 23.
93/43	B.W. Watson	A taxomomy of finite automata construction algorithms, p. 87.
93/44	B.W. Watson	A taxonomy of finite automata minimization algorithms, p. 23.
93/45	E.J. Luit J.M.M. Martin	A precise clock synchronization protocol,p.
93/46	T. Kloks D. Kratsch J. Spinrad	Treewidth and Patwidth of Cocomparability graphs of Bounded Dimension, p. 14.
93/47	W. v.d. Aalst P. De Bra G.J. Houben Y. Komatzky	Browsing Semantics in the "Tower" Model, p. 19.
93/48	R. Gerth	Verifying Sequentially Consistent Memory using Interface Refinement, p. 20.
94/01	P. America M. van der Kammen R.P. Nederpelt O.S. van Roosmalen H.C.M. de Swart	The object-oriented paradigm, p. 28.
94/02	F. Kamareddine R.P. Nederpelt	Canonical typing and Π-conversion, p. 51.
94/03	L.B. Hartman K.M. van Hee	Application of Marcov Decision Processe to Search Problems, p. 21.
94/04	J.C.M. Baeten J.A. Bergstra	Graph Isomorphism Models for Non Interleaving Process Algebra, p. 18.
94/05	P. Zhou J. Hooman	Formal Specification and Compositional Verification of an Atomic Broadcast Protocol, p. 22.
94/06	T. Basten T. Kunz J. Black M. Coffin D. Taylor	Time and the Order of Abstract Events in Distributed Computations, p. 29.
94/07	K.R. Apt R. Bol	Logic Programming and Negation: A Survey, p. 62.
94/08	O.S. van Roosmalen	A Hierarchical Diagrammatic Representation of Class Structure, p. 22.
94/09	J.C.M. Baeten J.A. Bergstra	Process Algebra with Partial Choice, p. 16.

94/10	T. verhoeff	The testing Paradigm Applied to Network Structure. p. 31.
94/11	J. Peleska C. Huizing C. Petersohn	A Comparison of Ward & Mellor's Transformation Schema with State- & Activitycharts, p. 30.
94/12	T. Kloks D. Kratsch H. Müller	Dominoes, p. 14.
94/13	R. Seljée	A New Method for Integrity Constraint checking in Deductive Databases, p. 34.
94/14	W. Peremans	Ups and Downs of Type Theory, p. 9.
94/15	R.J.M. Vaessens E.H.L. Aarts J.K. Lenstra	Job Shop Scheduling by Local Search, p. 21.
94/16	R.C. Backhouse H. Doornbos	Mathematical Induction Made Calculational, p. 36.
94/17	S. Mauw M.A. Reniers	An Algebraic Semantics of Basic Message Sequence Charts, p. 9.
94/18	F. Kamareddine R. Nederpelt	Refining Reduction in the Lambda Calculus, p. 15.
94/19	B.W. Watson	The performance of single-keyword and multiple-keyword pattern matching algorithms, p. 46.
94/20	R. Bloo F. Kamareddine R. Nederpelt	Beyond β-Reduction in Church's $\lambda\!\rightarrow$, p. 22.
94/21	B.W. Watson	An introduction to the Fire engine: A C++ toolkit for Finite automata and Regular Expressions.
94/22	B.W. Watson	The design and implementation of the FIRE engine: A C++ toolkit for Finite automata and regular Expressions.
94/23	S. Mauw and M.A. Reniers	An algebraic semantics of Message Sequence Charts, p. 43.
94/24	D. Dams O. Grumberg R. Gerth	Abstract Interpretation of Reactive Systems: Abstractions Preserving $\forall$CTL*, $\exists$CTL* and CTL*, p. 28.
94/25	T. Kloks	$K_{1,3}$-free and W_4-free graphs, p. 10.
94/26	R.R. Hoogerwoord	On the foundations of functional programming: a programmer's point of view, p. 54.
94/27	S. Mauw and H. Mulder	Regularity of BPA-Systems is Decidable, p. 14.
94/28	C.W.A.M. van Overveld M. Verhoeven	Stars or Stripes: a comparative study of finite and transfinite techniques for surface modelling, p. 20.
94/29	J. Hooman	Correctness of Real Time Systems by Construction, p. 22.
94/30	J.C.M. Baeten J.A. Bergstra Gh. Ştefanescu	Process Algebra with Feedback, p. 22.
94/31	B.W. Watson R.E. Watson	A Boyer-Moore type algorithm for regular expression pattern matching, p. 22.
94/32	J.J. Vereijken	Fischer's Protocol in Timed Process Algebra, p. 38.
94/33	T. Laan	A formalization of the Ramified Type Theory, p.40.
94/34	R. Bloo F. Kamareddine R. Nederpelt	The Barendregt Cube with Definitions and Generalised Reduction, p. 37.
94/35	J.C.M. Baeten S. Mauw	Delayed choice: an operator for joining Message Sequence Charts, p. 15.
94/36	F. Kamareddine R. Nederpelt	Canonical typing and Π-conversion in the Barendregt Cube, p. 19.
94/37	T. Basten R. Bol M. Voorhoeve	Simulating and Analyzing Railway Interlockings in ExSpect, p. 30.
94/38	A. Bijlsma C.S. Scholten	Point-free substitution, p. 10.

94/39	A. Blokhuis T. Kloks	On the equivalence covering number of splitgraphs, p. 4.
94/40	D. Alstein	Distributed Consensus and Hard Real-Time Systems, p. 34.
94/41	T. Kloks D. Kratsch	Computing a perfect edge without vertex elimination ordering of a chordal bipartite graph, p. 6.
94/42	J. Engelfriet J.J. Vereijken	Concatenation of Graphs, p. 7.
94/43	R.C. Backhouse M. Bijsterveld	Category Theory as Coherently Constructive Lattice Theory: An Illustration, p. 35.
94/44	E. Brinksma R. Gerth W. Janssen S. Katz M. Poel C. Rump J. Davies S. Graf B. Jonsson G. Lowe A. Pnueli J. Zwiers	Verifying Sequentially Consistent Memory, p. 160
94/45	G.J. Houben	Tutorial voor de ExSpect-bibliotheek voor "Administratieve Logistiek", p. 43.
94/46	R. Bloo F. Kamareddine R. Nederpelt	The λ-cube with classes of terms modulo conversion, p. 16.
94/47	R. Bloo F. Kamareddine R. Nederpelt	On Π-conversion in Type Theory, p. 12.
94/48	Mathematics of Program Construction Group	Fixed-Point Calculus, p. 11.
94/49	J.C.M. Baeten J.A. Bergstra	Process Algebra with Propositional Signals, p. 25.
94/50	H. Geuvers	A short and flexible proof of Strong Normalazation for the Calculus of Constructions, p. 27.
94/51	T. Kloks D. Kratsch H. Müller	Listing simplicial vertices and recognizing diamond-free graphs, p. 4.
94/52	W. Penczek R. Kuiper	Traces and Logic, p. 81
94/53	R. Gerth R. Kuiper D. Peled W. Penczek	A Partial Order Approach to Branching Time Logic Model Checking, p. 20.
95/01	J.J. Lukkien	The Construction of a small CommunicationLibrary, p.16.
95/02	M. Bezem R. Bol J.F. Groote	Formalizing Process Algebraic Verifications in the Calculus of Constructions, p.49.
95/03	J.C.M. Baeten C. Verhoef	Concrete process algebra, p. 134.
95/04	J. Hidders	An Isotopic Invariant for Planar Drawings of Connected Planar Graphs, p. 9.
95/05	P. Severi	A Type Inference Algorithm for Pure Type Systems, p.20.
95/06	T.W.M. Vossen M.G.A. Verhoeven H.M.M. ten Eikelder E.H.L. Aarts	A Quantitative Analysis of Iterated Local Search, p.23.
95/07	G.A.M. de Bruyn O.S. van Roosmalen	Drawing Execution Graphs by Parsing, p. 10.
95/08	R. Bloo	Preservation of Strong Normalisation for Explicit Substitution, p. 12.
95/09	J.C.M. Baeten J.A. Bergstra	Discrete Time Process Algebra, p. 20
95/10	R.C. Backhouse R. Verhoeven O. Weber	MathJpad: A System for On-Line Prepararation of Mathematical Documents, p. 15