A.S. Klusener
Completeness in real time process algebra

Computer Science/Department of Software Technology
Report CS-R9106 January

CWI, nationaal instituut voor onderzoek op het gebied van wiskunde en informatica
CWI is the research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a non-profit institute aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands organization for scientific research (NWO).
Completeness in Real Time Process Algebra

A.S. Klusener

CWI
P.O. Box 4079, 1009 AB Amsterdam,
The Netherlands
e-mail: stevenk@cwi.nl

Abstract

Recently, J.C.M. Baeten and J.A. Bergstra extended ACP with real time, resulting in a Real Time Process Algebra, called ACP*$\_r$ [BB89]. They introduced an equational theory and an operational semantics. Their paper does not contain a completeness result nor does it contain the definitions to give proofs in detail. In this paper we introduce some machinery and a completeness result.

The operational semantics of [BB89] contains the notion of an idle step reflecting that a process can do nothing more then passing the time before performing a concrete action at a certain point in time. This idle step corresponds nicely to our intuition but it results in uncountable branching transition systems.

In this paper we give a more abstract operational semantics, by abstracting from the idle step. Due to this simplification we can prove soundness and completeness easily. These results hold for the semantics of [BB89] as well, since both operational semantics induce the same equivalence relation between processes.

1985 Mathematics Subject Classification: 68Q10, 68Q40, 03G40, 06Q50.
Key Words & Phrases: Real Time, Process Algebra, ACP, Integration, SOS.
Note: This work is in part sponsored by ESPRIT Basic Research Action 3006, CONCUR.

Introduction

In order to reason algebraically with concurrent software systems, e.g. communication protocols and their specifications, process algebras such as CCS ([Mil80]), CSP ([Ho85]) and ACP ([BK84],[BW93]) are developed. Since some communication protocols contain timed constructs, such as timers and time-outs, there is a need to extend these process algebras with a notion of real time. Several authors have given a real time extension of a process algebra, e.g. see [MT90],[Wan90] for an extension of CCS, see [RR88] for an extension of CSP and see [BB89], [Gro90] for an extension of ACP. Nicollin and Sifakis have given a real time extension of a combination of CCS and ACP in [NS90]. Between these extensions several similarities and differences can be found. For example, most of these extensions are based on discrete time. However, the paper of Baeten and Bergstra is based on dense time and it introduces the interesting notion of integration, by which we can express that an action occurs somewhere within a (dense) interval. Hence, their work is more general and more complicated. In general their theory is undecidable, due to the notion of general integration over arbitrary subsets of $\mathbb{R}^\omega$. In this report we restrict ourselves to prefixed integration to tackle the complexity of integration over sets of reals. Prefixed integration requires that every action, which has a time variable $v$ as timestamp, is directly preceded by the binding integral of $v$. This restriction seems to be harmless from a practical point of view, since no "real-life" processes are known to us which are not covered by this restriction. Due to this simplification we can obtain a completeness result.

Real Time Process Algebra ([BB89]) concerns terms constructed from timed actions. A timed action is a combination of a symbolic action, taken from some alphabet $\mathcal{A}$, and a timestamp, taken from $\mathbb{R}^\omega$. This timestamp can be interpreted either absolute or relative. In absolute time every timestamp is
interpreted from the start time of the entire process (time zero) while in relative time every timesamp is interpreted from the point in time of the execution of the previous action. In this paper we will focus on absolute time only, in [BB89] a translation from relative to absolute time is given. Thus, the timed action \( a(2) \) denotes the atomic process which executes an action \( a \) at time \( 2 \), after which it terminates successfully. We expect some new identities that do not hold in standard Process Algebra ([EK84],[BW90]) such as

\[
a(2) \cdot (b(1) + c(3)) = a(2) \cdot c(3).
\]

After doing \( a(2) \) we have passed time \( 2 \) and in the following alternative composition \( b(1) + c(3) \) the first alternative cannot be chosen anymore and therefore it may be removed.

Furthermore, we consider integration as an essential feature of our theory. Integration is the alternative composition over a continuum of alternatives ([BB89]). An integral describes the interval in which a timed action may occur. For example, the process which executes an action \( c \) somewhere within the interval \( <0,1> \) is denoted by \( \int_{0<0,1>} a[v] \).

According to [BB89] we refer to Basic Real Time Process Algebra (the theory without parallelism) as \( \text{BPA}_\rho \). By adding integration we get \( \text{BPA}_\rho I \). Moreover, to Real Time Process Algebra (the theory with parallelism) is referred to as \( \text{ACP}_\rho \).

The development of such a theory implies the development of a system of axioms which generates exactly the required identities. To be sure that this system of axioms is exactly what we have in mind, we give an operational semantics. In our case an operational semantics assigns to every process expression a transition system which reflects the behaviour of this process. A transition system is a directed acyclic graph in which each arrow is labelled with a timed action. These transitions systems are generated by giving a Transition System Specification in Structured Operational Semantics according to Plotkin ([Plot81]). We define an equivalence notion (which must be a congruence) on these transition systems by defining a bisimulation relation ([Mil89],[Par81]). If we have defined such an operational semantics we can validate the axiom system by proving soundness and completeness.

In [BB89] an axiom system is given together with an operational semantics which yields uncountable transition systems even for terms without integration and recursion. In that paper, neither soundness nor completeness are treated thoroughly.

In this paper we will give an abstract operational semantics which yields finite transition systems for terms without integration and recursion. The action rules defining this abstract semantics are similar to the action rules for the standard operational semantics of \( \text{ACP}_\rho \) as given in [Gla87]. We prove soundness and completeness of this abstract operational semantics with respect to Real Time Process Algebra. We can extend these results to the semantics of [BB89] since we prove that both semantics are equivalent in the sense that they induce the same identities between terms.

In this paper we define the notion of a basic term. A basic term has very nice properties, like a strong correspondence between a basic term as its transition system. Hence, we can prove completeness easily for this subset of terms. Since we prove as well that every term is derivably equal to a basic term within the theory, which is sound, we can extend the completeness result to all transition systems which can be represented by a term.

We do not consider recursion in this paper.

Contents

1 The Original Operational Semantics 3
2 The Theory of Basic Real Time Process Algebra 6
  2.1 BPA_\rho 6
  2.2 Basic Terms 8
1 The Original Operational Semantics

In this section we give some intuition for timed processes by introducing the operational semantics of [BB89] for process expressions over Basic Real Time Process Algebra (BPA_{\delta}). We do not yet consider integration in this section. Let \( A_{\delta} \) be the set of actions, containing the constant \( \delta \). The alphabet of the theory BPA_{\delta} is

\[ A_{\delta}^{\text{time}} = \{ a(t) | a \in A_\delta, t \in \mathbb{R}^{\geq 0} \} \]

Similarly we use \( A^{\text{time}} \), as the set of timed actions without timed \( \delta \)'s. In the sequel we refer to actions from \( A_{\delta} \) as symbolic actions and we refer to actions from \( A_{\delta}^{\text{time}} \) as timed actions. Moreover, process expressions are simply called terms. The set \( T \) of closed terms over \( \text{BPA}_{\delta} \) is generated by the alphabet \( A_{\delta}^{\text{time}} \) and the binary operators + for alternative composition and \( \cdot \) for sequential composition and the operator \( \gg \), called the \( (\text{absolute}) \) time shift. In the previous sentence we referred
to closed terms as terms without recursion variables. In this report we do not consider recursion, hence if we consider a term it is meant to be a term which is closed w.r.t. recursion variables. In the sequel we will introduce another kind of variables, called time variables.

The (absolute) time shift, $\rightarrow$, takes a nonnegative real number and a process term; $t \rightarrow X$ denotes that part of $X$, which starts after $t$. The set $T$ with typical elements $p, p_1, p_2$ is defined in the following way, where $a \in A_6$ and $r \in \mathbb{R}^+$:

$$p \in T \quad p := a(r) \mid p_1 + p_2 \mid p_1 \cdot p_2 \mid r \rightarrow p$$

The semantics of [BB89] assigns to every term (in $T$) a transition system in which each state is a pair consisting of a term and a point in time and in which each transition is labelled by a timed (non $\delta$) action. Within this semantics each transition system concerns three relations

- **Step** $\subseteq (T \times \mathbb{R}^0) \times A_{\text{time}} \times (T \times \mathbb{R}^0)$
- **Idle** $\subseteq (T \times \mathbb{R}^0) \times (T \times \mathbb{R}^0)$
- **Terminate** $\subseteq (T \times \mathbb{R}^0) \times A_{\text{time}} \times \mathbb{R}^0$

These three relations are defined as the least relations satisfying the Plotkin rules given in this section. We write

- $<x,t,a(r)> <z',t'>$ for $(<x,t,a(r)>,<z',t'>) \in \text{Step}$
- $<x,t,\rightarrow> <z',t'>$ for $(<x,t>,<z',t'>) \in \text{Idle}$
- $<x,t,a(r)> <\sqrt{,}t'>$ for $(<x,t,a(r),t'>) \in \text{Terminate}$

We always have $t' = r$ in Step and Terminate and $x \equiv z'$ in Idle. The term $a(1)$ denotes the process which performs an action at time $1$, after which it is successfully terminated. The root node of the transition system of $a(1)$ is $<a(1),0>$, denoting that time starts at zero. In Table 1 the action rules for timed actions and the alternative composition are given ($a$ is taken from $A$).

<table>
<thead>
<tr>
<th>$t &lt; r$</th>
<th>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt;a(r),t&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$t &lt; s &lt; r$</td>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;a(r),t&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t&gt;a(r) &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t&gt;a(r) &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t&gt;a(r) &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$t &lt; s &lt; r$</td>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;a(r),t&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
<tr>
<td>$&lt;x,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
<td>$&lt;x+y,t,a(r)&gt; &lt;\sqrt{,}r&gt;$</td>
</tr>
</tbody>
</table>

Table 1: Action Rules for Atomic Actions and Alternative Composition

From $<a(1),0>$ an idle transition is possible to a state of the form $<a(1),t>$ with $0 < t < 1$. An idle transition is a transition in which only the time component is increased without executing any action. In general, from each state $<a(1),t>$ an idle transition is possible to $<a(1),t'>$, whenever $t < t' < 1$. Further from each state $<a(1),t>$ a a(1)-transition to $<\sqrt{,}1>$ is possible whenever $t < 1$.

This transition system of the term $a(1)$ can be represented by the left-hand process diagram given in Figure 1. A process diagram is simply a pictorial representation of a transition system. It is not possible to make a picture of the transition system itself, since it has uncountably many transitions. The intuition behind such a process diagram is that the process starts in the top-point. It can idle by going to a lower point without crossing any line, whereas the execution of an action $a$ at time $r$ is
reflected by going to a dashed line at level r labelled with a. Only dashed lines may be crossed, after landing on them.

A very particular set of atomic actions is the set of δ(r)-terms. δ(1) can do nothing more than idling until 1. Thus the root node is <δ(1),0> and from each state <δ(1),t> an idle transition to <δ(1),t'> is possible, whenever t < t' < 1. The action rules defining the transition systems of timed actions are given in Table 1. In Table 1 the rules for the alternative composition are given as well; the transition system of p + q is defined in terms of the transition systems of p and q. The behaviour of p + q can be considered as the "union" of the behaviour of p and that of q.

The transition system of a(1) + b(2) can be represented by the process diagram given in Figure 2.

A state μ (in Figure 2) is of the form <a(1) + b(2), t> with 0 ≤ t < 1. From μ both a terminating a(1)-transition to <√2, 1> and a terminating b(2)-transition to <√2, 2> are possible. However, from a state like ν of the form <a(1) + b(2), t> with 1 ≤ t < 2 only a terminating b(2)-transition to <√2, 2> is possible. Hence, by idling from <a(1) + b(2), t₀> to <a(1) + b(2), t₁> with 0 ≤ t₀ < 1 ≤ t₁ < 2 we have lost the possibility of executing the a(1)-summand. Thus one could say that a choice has been made at time 1; after the choice has been made for b(2) the summand a(1) has become redundant.

The transition system of a(1) + δ(1) consists of exactly the same relations as the transition system of a(1). The summand δ(1) contributes only idle steps which are contributed by the summand a(1) as well.

However if we consider a(1) + δ(2), the δ(2) summand contributes idle transitions which are not contributed by a(1), since δ(2) has idle transitions to points in time between 1 and 2. The transition system of a(1) + δ(2) can be represented by the process diagram on the right-hand side in Figure 2. The action rules for sequential composition, are given in Table 1.

The last operator we introduce is the (absolute) time shift denoted by ⇒, which takes a real number s and a process X and delivers that part of X which starts after s. Hence, before s it can only idle or δ a transition to a state after s. The action rules for the (absolute) time shift operator are
given in Table 1 as well. Note that we can show by induction of the length of the derivation that 
< z, t > \xrightarrow{a(r)} < x', t' > implies that t < r and t' = r. Or, the execution of an action a at time r is possible only in a state with a time component smaller than r.

## 2 The Theory of Basic Real Time Process Algebra

### 2.1 BPA_δ

BPA_δ is the theory of Basic Real Time Process Algebra ([BB89]). It consists of A1 - A5, which are the standard axioms of Basic Process Algebra, extended by some axioms stating the specific real time properties and defining the (absolute) time shift. The axioms of BPA_δ are given in Table 3, we abbreviate δ(0) by δ. Notice that the laws ATA2 and ATA4 are generalisations of the BPA_δ laws A6 and A7.

Using BPA_δ we can prove:

\[
\begin{align*}
5 & \Rightarrow (a(4) + b(6) + c(7) \cdot d(8)) = b(6) + c(7) \cdot d(8) \\
5 & \Rightarrow (a(4) + b[3]) = δ(5) \\
δ(1) + a(2) \cdot b(3) + δ(3) \cdot c(4) & = a(2) \cdot b(3) + δ(3) \\
a(0) + b(2) \cdot (c(1) + c(3)) + δ(3) \cdot c(2) & = b(2) \cdot c(3) + a(3) \cdot δ(3)
\end{align*}
\]

### Some Definitions and Properties

We introduce some definitions of inductive functions on terms; \(U(p)\) is the ultimate delay which is the smallest timestamp which is not reachable by the process \(p\) by idling. \(\text{initim}(p)\) is the set of points in time at which \(p\) can execute an initial action. \(L(p)\) is the latest moment at which \(p\) can execute an action.

**Definition 2.1.1** Definition of the syntactic operations \(U(p), \text{initim}(p), L(p)\).
\[ X + Y = Y + X \]
\[ (X + Y) + Z = X + (Y + Z) \]
\[ X + X = X \]
\[ (X + Y) \cdot Z = X \cdot Z + Y \cdot Z \]
\[ (X \cdot Y) \cdot Z = X \cdot (Y \cdot Z) \]

\[ BPA = A1 - A5 \]

\begin{align*}
\text{ATA1} & \quad a(0) = \delta \\
\text{ATA2} & \quad \delta(t) \cdot X = \delta(t) \\
\text{ATA3} & \quad \delta(t) + \delta(r) = \delta(r) \\
\text{ATA4} & \quad a(t) + \delta(t) = a(t) \\
\text{ATA5} & \quad a(t) \cdot X = a(t) \cdot (t \gg X) \\
\text{ATB1} & \quad \not< r \quad \therefore a(r) = a(r) \\
\text{ATB2} & \quad \not\geq r \quad \therefore a(r) = \delta(t) \\
\text{ATB3} & \quad \not\gg (X + Y) = (t \gg X) + (t \gg Y) \\
\text{ATB4} & \quad \not\gg (X \cdot Y) = (t \gg X) \cdot Y
\end{align*}

Table 3: BPAɛ = BPA + ATA1-5 + ATB1-4

\[ a \in A \]
\[ U(\delta(t)) = t \]
\[ U(a(t)) = t \]
\[ U(X \cdot Y) = U(X) \]
\[ U(X + Y) = \max[U(X),U(Y)] \]
\[ U(r \gg X) = \max[U(X),t] \]

\[ \text{inittim}(\delta(t)) = \emptyset \]
\[ \text{inittim}(a(t)) = \begin{cases} \\
\emptyset & \text{if } t = 0 \\
\{t\} & \text{otherwise} \\
\end{cases} \]
\[ \text{inittim}(X \cdot Y) = \text{inittim}(X) \]
\[ \text{inittim}(X + Y) = \text{inittim}(X) \cup \text{inittim}(Y) \]
\[ \text{inittim}(r \gg X) = \{t | t \in \text{inittim}(X) \land t > r\} \]

\[ L(X) = \max(\text{inittim}(X)) \text{ where } \max(\emptyset) = 0 \]

The ultimate delay is introduced already in [BB89], however here it is not defined as a function on the syntax but as an operator within the theory. The other functions are new here. The size of a term \( X \) is defined as the number of operators in the term \( X \).

Terms can be considered equal in different ways. Therefore we need some different notations. Two terms \( p, q \) are syntactically equivalent, denoted by \( p \equiv q \), if they are constructed in exactly the same way from the atomic actions and the constructors. Normally we are not much interested whether two terms are syntactically equivalent, but more whether they are equal modulo the axioms A1 and A2, denoted by \( p \simeq q \). Sometimes we refer to this equivalence as "the terms \( p \) and \( q \) have the same form". If there is a derivation in the theory BPAɛ connecting two terms \( p \) and \( q \), then they are called
derivably equal to each other, denoted by \( p \equiv q \). Finally, we have two notions of summand inclusion, the idea is that \( p \) is a summand of \( p + q \). We use derivable summand inclusion, if we consider the terms in the context of the theory. However, if we want to consider the form of the terms only, for example when we compare different semantics, we need a notion of syntactic summand inclusion.

**Definition 2.1.2**

1. **Syntactic equivalence**
   \( p \equiv q \)
2. **Syntactic equivalence modulo \( A_1 \) and \( A_2 \)**
   \( A_1, A_2 \vdash p = q \)
3. **Derivability in \( BPA_\delta \)**
   \( BPA_\delta \vdash p = q \)
4. **Derivable summand inclusion**
   \( BPA_\delta \vdash p + q = q \)
5. **Syntactic summand inclusion**
   \( \subseteq \) is the least relation satisfying:
   \[ p \subseteq q \implies p \subseteq q' \]
   \[ p \subseteq q \implies p \subseteq q + q' \]

For convenience we generalise the definition of \( \simeq \) by adding \( p \simeq p + \delta \) and taking the transitive closure. Note that \( \delta \) is an abbreviation of \( \delta(0) \).

Now we can state:

\[
\begin{align*}
U(p) \leq t & \iff a(t) \cdot p = c(t) \cdot \delta(t) \\
p = q & \implies \text{active}(p) = \text{active}(q) \land \text{init}(p) = \text{init}(q) \\
U(t \Rightarrow p) & = \max(t, U(p))
\end{align*}
\]

### 2.2 Basic Terms

If we consider terms like \( a(5) \cdot b(6) + \delta(1) \) and \( a(5) \cdot (b(6) + c(4)) \), we may say that both of them contain “redundant” parts; parts that can be removed by application of the axioms of \( BPA_\delta \). Within \( BPA_\delta \), both terms are derivably equal to the term \( c(5) \cdot b(6) \), which does not contain any redundant parts. Hence, this last term can be seen as a kind of normal form and in the sequel we will call it a basic term.

\( B \) is the set of basic terms. To define this set of basic terms we first define auxiliary sets \( B^r(t) \) with \( r \in \mathbb{R}_\omega^0 \) and \( t \in \mathbb{R}_\omega^0 \), where \( \mathbb{R}_\omega^0 = \mathbb{R}_\omega^0 \cup \{\omega\} \). We extend the standard ordering \( < \) in \( \mathbb{R}_\omega^0 \) to an ordering on \( \mathbb{R}_\omega^0 \) by putting \( s < \omega \). \( p \in B^r(t) \) means that \( p \) is a basic term which can idle until \( t \) without losing alternatives and it has a deadlock at \( r \) whenever \( r \neq \omega \). Hence, we have \( r > \omega \) in case \( B^r(t) \neq \emptyset \). Moreover, \( B(t) \) is the set of basic terms starting after \( t \), regardless of a possible deadlock.

**Definition 2.2.1** Let \( p, q \in T, a \in A, t \in \mathbb{R}_\omega^0 \) and \( r, s,\in \mathbb{R}_\omega^0 \).

\( B, B(t) \) and \( B^r(t) \) are the smallest sets satisfying:

\[
\begin{align*}
\text{Let } t < s & \implies a(s) \in E^r(t) \\
\text{Let } r < s & \implies \delta(s) \in E^r(t) \\
\text{Let } t < s \land (p \in E^r(s) \lor p \simeq \delta) & \implies a(s) \cdot p \in E^r(t) \\
p \in B^r(t) \land q \in B^r(t) \land (r = z \lor (r < z = \omega \land r > U(q))) & \implies \begin{cases} 
\text{Let } p + q, q + p & \in E^r(t) \\
p \in B^r(t) & \implies p + \delta, \delta + p & \in E^r(t)
\end{cases}
\end{align*}
\]

\[
\begin{align*}
B(t) & = \cup_r B^r(t) \\
B & = B(0) \cup \{\delta\}
\end{align*}
\]

Note that basic terms have the following properties...
2.2 Basic Terms

Lemma 2.2.2

\[ t \leq t' \;
\iff \quad B(t) \supseteq B(t') \]

\[ a(t) \cdot p \in B \land p \neq \delta \quad \iff \quad p \in B(t) \]

\[ \delta(t) \subseteq p \land t > 0 \quad \iff \quad t = U(p) > L(p) \]

Proof. Direct by the definition of basic terms

By definition of basic terms we know that only prefix multiplication is used, and the (absolute) time shift operator \( \gg \) does not occur in a basic term. In the sequel we abbreviate terms of the form \( p_1 + \ldots + p_n \) by \( \sum_{1 \leq i \leq n} p_i \) or \( \sum_{i \in \{1, \ldots, n\}} p_i \). As is usual we use the convention \( \sum_{i \in \emptyset} p_i \simeq \delta \). Hence, every basic term is of the form

\[
\sum_{1 \leq i \leq n} a_i(t_i) \cdot p_i + \sum_{1 \leq j \leq m} b_j(r_j)
\]

where \( n + m > 0 \), \( p_i \in E \) and \( a_i \) and \( b_j \) are atomic actions. The addition of terms like \( p + \delta \) to \( B \) has technical reasons. Due to this addition we can have empty summations within a basic term.

Moreover, by the previous Lemma we know:

\[
\forall i \quad a_i \neq \delta \land (p_i \in B(t_i) \lor p_i \simeq \delta) \]

\[
U(p) > L(p) \quad \iff \quad \exists j \quad b_j = \delta \land r_j = U(p)
\]

Examples of basic terms:

\[
\forall t < 5 \quad b(5) \quad \in \quad B^{\sim(t)}
\]

\[
\forall r < 10 \quad \delta(10) \quad \in \quad B^{10(r)}
\]

\[
\forall s < 5 \quad b(s) + \delta(10) \quad \in \quad B^{10(s)}
\]

\[
\forall u < 2 \quad a(2) \cdot (b(5) + \delta(10)) \quad \in \quad B^{\sim(u)}
\]

Lemma 2.2.3 For each \( p, q \in B \) there is a \( z \in B \) such that \( BPAp \delta + z = p \cdot q \)

Proof. Assume

\[ p \simeq \sum_{i \in I} a_i(t_i) \cdot p_i + \sum_{j \in J} b_j(s_j) \]

\[ q \simeq \sum_{k \in K} c_k(t_k) \cdot q_k + \sum_{l \in L} d_l(u_l) \]

We take

\[
K^r = \{ k \in K | t_k > r \}
\]

\[
L^s = \{ l \in L | u_l > s \}
\]

Construct \( \bar{p} \) from \( p \) by removing the \( \delta(t) \) summands, hence

\[
\bar{p} \simeq \sum_{i \in I} a_i(t_i) \cdot p_i + \sum_{j \in J} b_j(s_j)
\]

Now, we can proof the Lemma by induction on the depth of \( p \).

- \( I = \emptyset \) and \( J \neq \emptyset \)
  Construct first \( \bar{z} \) such that \( \bar{z} = \bar{p} \cdot q = \sum_{j \in J} b_j(s_j) \cdot (s_j \gg q) \) by taking

\[
\bar{z} \simeq \sum_{j \in J} b_j(s_j) \cdot \left\{ \sum_{k \in K^r} c_k(t_k) \cdot r_k + \sum_{l \in L^s} d_l(u_l) \right\}
\]

Then we can construct \( z \) as follows:

\[
z \simeq \begin{cases} \bar{z} + U(p) \in B & \text{if } U(p) > U(\bar{p}) \\ \bar{z} \in B & \text{otherwise} \end{cases}
\]
• $I \neq \emptyset$

\[ \bar{p} \cdot q \simeq (\sum_{i \in I} a_i(r_i) \cdot p_i + \sum_{j \in J} b_j(s_j)) \cdot q \]
\[ = \sum_{i \in I} (a_i(r_i) \cdot p_i) \cdot q + \sum_{j \in J} b_j(s_j) \cdot q \]
\[ = \sum_{i \in I} a_i(r_i) \cdot (r_i \Rightarrow (p_i \cdot q)) + \sum_{j \in J} b_j(s_j) \cdot (s_j \Rightarrow q) \]

Since $p_i \in B(r_i)$ we know $r_i \Rightarrow (p_i \cdot q) = (r_i \Rightarrow p_i) \cdot q = p_i \cdot q$.

By induction there is for each $i \in I$ a $z_i \in B$ such that $z_i = q_i \cdot q$.

Hence, by constructing $\bar{z}$ as follows we have $\bar{z} = \bar{p} \cdot q$

\[ \bar{z} \simeq \sum_{i \in I} a_i(r_i) \cdot z_i + \sum_{j \in J} b_j(s_j) \cdot \{\sum_{k \in K} c_k(t_k) \cdot r_k + \sum_{l \in L} d_l(u_l)\} \]

And we can construct $z \in B$ satisfying $z = p \cdot q$ by taking

\[ z = \begin{cases} \bar{z} + \delta(U(p)) \in B & \text{if } U(p) > U(\bar{p}) \\ \bar{z} \in B & \text{otherwise} \end{cases} \]

Lemma 2.2.4 For each term $p \in T$ there is a basic term $p_b$ such that $\text{BPA}_{\delta} \vdash p = p_b$


1. $p \equiv a(t)$
   If $t = 0$
   Then
   \[ p \equiv a(0) \overset{\text{ATA1}}{=} \delta \in B \]
   Else
   \[ p \equiv a(t) \in B \]

2. $p \equiv q \cdot r$

   By induction there are $q_b, r_b \in B$ such that $\text{BPA}_{\delta} \vdash q = q_b$ and $\text{BPA}_{\delta} \vdash r = r_b$ and the case follows by the previous Lemma.

3. $p \equiv q + r$

   By induction there are $q_b, r_b \in B$ such that $\text{BPA}_{\delta} \vdash q = q_b$ and $\text{BPA}_{\delta} \vdash r = r_b$, say

   \[ q_b \simeq \sum_{i \in I} a_i(r_i) \cdot q_i + \sum_{j \in J} b_j(s_j) \]
   \[ r_b \simeq \sum_{k \in K} c_k(t_k) \cdot r_k + \sum_{l \in L} d_l(u_l) \]

   Construct $\bar{q}_b$ and $\bar{r}_b$ from $q_b$ and $r_b$ by removing the $\delta(t)$ summands, hence

   \[ \bar{J} = \{ j \in J \mid b_j \neq \delta \} \]
   \[ \bar{L} = \{ l \in L \mid d_l \neq \delta \} \]
   \[ \bar{q}_b \simeq \sum_{i \in I} a_i(r_i) \cdot q_i + \sum_{j \in J} b_j(s_j) \]
   \[ \bar{r}_b \simeq \sum_{k \in K} c_k(t_k) \cdot r_k + \sum_{l \in L} d_l(u_l) \]

   And we can conclude

   \[ p = \begin{cases} \bar{q}_b + \bar{r}_b + \delta(U(q_b + r_b)) \in B & \text{if } U(q_b + r_b) > U(\bar{q}_b + \bar{r}_b) \\ \bar{q}_b + \bar{r}_b \in B & \text{otherwise} \end{cases} \]
4. $p \equiv t \succ q$

By induction there is a $q_b \in B$ such that $BPA_{\rho \delta} \vdash q = q_b$, say

$$q_b \simeq \sum_{i \in I} a_i(r_i) \cdot q_i + \sum_{j \in J} b_j(s_j)$$

Define $I^r$ and $J^r$ as

$$I^r = \{ i \in I | r_i > r \}$$
$$J^r = \{ j \in J | s_j > r \}$$

Now

$$p = \begin{cases} \varepsilon(t) \in B & \text{if } I^r \cup J^r = \emptyset \\ \sum_{i \in I^r} a_i(r_i) \cdot q_i + \sum_{j \in J^r} b_j(s_j) \in B & \text{otherwise} \end{cases}$$

3 An Abstract Operational Semantics for $BPA_{\rho \delta}$

3.1 An Abstraction from the idle transition

In the first section the operational semantics of [BB89] is presented. In this operational semantics every transition system has uncountably many idle transitions. In this section we abstract from these idle transitions; we redefine the operational semantics such that there are no idle transitions anymore and hence all transition systems of finite closed $BPA_{\rho \delta}$ terms have a finite number of transitions.

The transition system of $a(1)$ will now contain only one labelled arrow as given in the left-hand side of Figure 3. This new semantics is analogous to the semantics for $ACP$ presented in [Gla87], where the transition system of $a$ is simply:

$$a \xrightarrow{a} \checkmark$$

Moreover, in the same semantics the transition system of $a \cdot p$ contains the transition

$$a \cdot p \xrightarrow{a} p$$

In a real time setting we have to take the time stamps into account. In $a(r) \cdot p$, after doing the $a(r)$ action, only that part of $p$ can be done which starts after $r$, which is denoted by $r \succ p$. Hence, the transition system of $a(r) \cdot p$ contains the transition

$$a(r) \cdot p \xrightarrow{a(r)} r \succ p$$

If $p$ can perform an action $b(t)$ then $s \succ p$ can perform this action only when $t > s$.

In Figure 3 the “abstract” transition systems of the terms $a(1)$ and $a(1) + b(2)$ are given, together with the corresponding process diagrams. We have to be careful with deadlocks, because we have to distinguish the term $\delta(1)$ from $\delta(2)$. In the next section we add an extra condition to the definition of bisimulation, saying that only nodes with the same ultimate delay are bisimilar. Since states are now terms from $T$, every state has an ultimate delay. In a later section we will introduce a relation $Deadlock \subseteq T \times \mathbb{R}^{\geq 0}$, like the relations $Step$ and $Terminate$, where $(p, r) \in Deadlock$ means that $p$ has a deadlock at $r$. 
3.2 An Abstract Semantics with an Extended Notion of Bisimulation

Now we can give the more formal definition in Structured Operational Semantics (see Table 4). In this abstract semantics every state is a term from \( T \) and every transition is labelled by a timed atomic action \( \neq \delta \). The operational semantics given in this section concerns two relations:

\[
\begin{align*}
\text{Step} & \subseteq T \times A^{\text{time}} \times T \\
\text{Terminate} & \subseteq T \times A^{\text{time}}
\end{align*}
\]

These relations are defined as the least relations satisfying the action rules of Table 4. We write:

\[
\begin{align*}
p \xrightarrow{a(r)} p' & \text{ for } (p, a(r), p') \in \text{Step} \\
p \xrightarrow{a(r)} \checkmark & \text{ for } (p, a(r)) \in \text{Terminate}
\end{align*}
\]

Of course, it would be possible to define \( \text{State} \) as \( T \cup \{\checkmark\} \), in which case the relation \( \text{Terminate} \) would not be needed anymore. However, in this section we follow the convention of [BB99].

If we say that the term \( p \) has a certain transition, we mean that the transition system associated to \( p \) has this transition.

We have to adapt the definition of strong bisimulation, since \( p \xrightarrow{a(r)} \checkmark \) is only an abbreviation of \( (p, a(r)) \in \text{Terminate} \) and we need an extra condition, concerning the ultimate delay of two related terms. We formulate the following definition:

**Definition 3.2.1 ultimate delay bisimulation.**

Let \( p, q \in T \), then \( p \leftrightarrow q \iff \) there is a relation \( R \subseteq T \times T \) such that:

1. the roots \( p \) and \( q \) are related by \( R \)

2. if \( s \xrightarrow{a(r)} s' \) is a transition in the transition system of \( p \) and \( R(s, t) \)
   then there is a transition \( t \xrightarrow{a(r)} t' \) in the transition system of \( q \) such that \( R(s', t') \)

3. if \( t \xrightarrow{a(r)} t' \) is a transition in the transition system of \( q \) and \( R(s, t) \)
   then there is a transition \( s \xrightarrow{a(r)} s' \) in the transition system of \( p \) such that \( R(s', t') \)
3.2 An Abstract Semantics with an Extended Notion of Bisimulation

\[
\begin{align*}
\text{atom} & : a(r) \xrightarrow{a(r)} \sqrt{ } \\
\rightarrow & : \quad \frac{p \xrightarrow{a(r)} \sqrt{ }}{p \cdot q \xrightarrow{a(r)} r \Rightarrow q} \quad \frac{p \xrightarrow{a(r)} p'}{p \cdot q \xrightarrow{a(r)} p' \cdot q} \\
\Rightarrow & : \quad \frac{p \xrightarrow{a(r)} \sqrt{ }}{p + q \xrightarrow{a(r)} \sqrt{ } , \quad q + p \xrightarrow{a(r)} \sqrt{ }} \\
& \quad \frac{p \xrightarrow{a(r)} p'}{p + q \xrightarrow{a(r)} p' , \quad q + p \xrightarrow{a(r)} p'}
\end{align*}
\]

Table 4: Transition System Specification for BPAρδ

4. If \( s \xrightarrow{a(r)} \sqrt{ } \) is a terminating transition in the transition system of \( p \) and \( R(s, t) \), then there is a terminating transition \( t \xrightarrow{a(r)} \sqrt{ } \) in the transition system of \( q \)

5. If \( t \xrightarrow{a(r)} \sqrt{ } \) is a terminating transition in the transition system of \( q \) and \( R(s, t) \), then there is a terminating transition \( s \xrightarrow{a(r)} \sqrt{ } \) in the transition system of \( q \)

6. If \( s \) and \( t \) are related, then they must have the same ultimate delay

Lemma 3.2.2 \( \forall p \in T \forall r \in \mathbb{R}_{\geq 0} \)

\[
r \in \text{init}(p) \iff \exists a \left( \exists p' \quad p \xrightarrow{a(s)} p' \lor p \xrightarrow{a(r)} \sqrt{ } \right)
\]

Proof. 
⇒ By induction on the size of \( p \).
⇐ By induction on the length of the derivation. \( \square \)

Lemma 3.2.3 \( \forall p, q \in T \)

\[
p \xrightarrow{a} uq \implies U(p) = U(q) \land L(p) = L(q)
\]

Proof. Assume \( p \xrightarrow{a} uq \), then \( U(p) = U(q) \) follows by definition. \( L(p) = L(q) \) follows from the previous Lemma and the definition of \( \xrightarrow{a} u \) since \( L(p) = \max(\text{init}(p)) \) \( \square \)

An axiom is sound w.r.t. this operational semantics if two arbitrary terms which are instantiations of the axiom have bisimilar transition systems, according to the definition 3.2.1. In the next theorem the
soundness of the axioms of \( \text{BPA}_\delta \) is stated. Note that for the soundness of the theory \( \text{BPA}_\delta \) itself, we need that bisimulation equivalence is a congruence as well. In the sequel we extend the theory \( \text{BPA}_\delta \) resulting in new soundness theorems, but only in this case we provide a formal proof. Hence, the proof of the next theorem serves as as an example how to prove the soundness of a set of axioms w.r.t. to an operational semantics.

**Theorem 3.2.4** The axioms of \( \text{BPA}_\delta \) are sound w.r.t. to the ultimate delay bisimulation equivalence \( \leftrightarrow \).

**Proof.** See Appendix A

3.3 An Abstract Operational Semantics with \( \delta \)-Rules

The operational semantics of [BB89] has the notion of an *idle* transition. In this operational semantics every state consists of an expression part and a time part; an *idle* transition increases the time part while the expression part remains the same. In this way a possible deadlock is reflected; the transition system of \( a(2) + \delta(5) \) has uncountably many *idle* transitions before 5. In the operational semantics presented in the previous section we abstract from these *idle* transitions by adding an extra condition to the notion of bisimulation saying that bisimilar terms must have the same ultimate delay. In this way, terms are distinguished as well if they differ only in their deadlock behaviour.

Another possibility is to introduce a relation

\[
\text{Deadlock} \subseteq T \times \mathbb{R}^>0
\]

containing all pairs \((p, r)\) such that \(p\) has a deadlock at time \(r\). Hence, \(\{(\delta(5), 5), (6 \not\geq b(4), 6)\} \subseteq \text{Deadlock}\). We write

\[
p \xrightarrow{\delta(r)} \checkmark \quad \text{for} \quad (p, r) \in \text{Deadlock}
\]

or we say "\(p\) has a \(\delta(r)\)-transition" if we are not interested in the timestamp of the deadlock. Since terms differing only in their deadlock behaviour have different \(\delta\)-transitions now, we do not need anymore the extra condition concerning the ultimate delay of related terms. The notions of \(\text{State}, \text{Label}\) and the definitions of \(\text{Step}\) and \(\text{Terminate}\) remain as in the previous section. Also the rules for \(a \not\in \delta\) remain the same as in the previous section, therefore we may "import" these rules. A \(\delta\)-rule is an action rule which generates \(\delta\)-transitions. A term has a possible deadlock whenever its ultimate delay \(U(p)\) is later than the latest possible action, denoted by \(L(p)\). The rules are given in Table 5.

<table>
<thead>
<tr>
<th>Import Table 4</th>
</tr>
</thead>
</table>
| \[
U(p) > L(p) \\
\frac{p \xrightarrow{\delta(U(p))} \checkmark}
| Table 5: Transition System Specification for \( \text{BPA}_\delta \) with \( \delta \)-rule |

**Definition 3.3.1** *strong bisimulation*

Let \(p, q \in \text{State}\) then \(p \leftrightarrow q\) iff there is a relation \(R \subseteq \text{State} \times \text{State}\) such that:
3.3 An Abstract Operational Semantics with $\delta$-Rules

1. the roots $p$ and $q$ are related by $R$

2. if $s \xrightarrow{a(s)} s'$ is a transition in the transition system of $p$ and $R(s, t)$ then there is a transition $t \xrightarrow{a(t)} t'$ in the transition system of $q$ such that $R(s', t')$

3. if $t \xrightarrow{a(t)} t'$ is a transition in the transition system of $q$ and $R(s, t)$ then there is a transition $s \xrightarrow{a(s)} s'$ in the transition system of $p$ such that $R(s', t')$

4. if $s \xrightarrow{a(s)} \top$ is a terminating transition in the transition system of $p$ and $R(s, t)$ then there is a terminating transition $t \xrightarrow{a(t)} \top$ in the transition system of $q$

5. if $t \xrightarrow{a(t)} \top$ is a terminating transition in the transition system of $q$ and $R(s, t)$ then there is a terminating transition $s \xrightarrow{a(s)} \top$ in the transition system of $p$

6. if $s \xrightarrow{\delta(s)} \top$ is a deadlock transition in the transition system of $p$ and $R(s, t)$ then there is a deadlock transition $t \xrightarrow{\delta(t)} \top$ in the transition system of $q$

7. if $t \xrightarrow{\delta(t)} \top$ is a deadlock transition in the transition system of $q$ and $R(s, t)$ then there is a deadlock transition $s \xrightarrow{\delta(s)} \top$ in the transition system of $p$

Note that this additional $\delta$ rule induces the relation Deadlock $\subseteq T \times \mathbb{R}^2$. Since we import the action rules of Table 4 as well, the relations Step and Terminate are imported as well. Next, we give some simple Lemma's.

Lemma 3.3.2 $\forall p \in T \forall u \in \mathbb{R}^2$

$$p \xrightarrow{U(u)} \top \iff U(p) = u \land U(p) > L(p)$$

Proof. Directly from the action rules for $\delta$ transitions. $\square$

Lemma 3.3.3 $\forall p, q \in T$

$$p \iff q \implies U(p) = U(q) \land L(p) = L(q)$$

Proof. Assume $p \iff q$. $L(p) = L(q)$ follows from the Lemma 3.2.2 and Definition 3.3.1. If $U(p) > L(p)$ then $p \xrightarrow{U(p)} \top$ and by Definition 3.3.1 $q \xrightarrow{U(q)} \top$ as well and we may conclude $U(p) = U(q) > L(q)$. If $U(p) = L(p)$ then $U(p) = U(q)$ follows from $L(p) = L(q)$ and $U(p) > L(p) \iff U(q) > L(q)$. $\square$

Lemma 3.3.4 Correspondence Lemma for basic terms w.r.t. the operational semantics of Table 5

$\forall p \in B \forall a \in A \forall r \in \mathbb{R}^2$

$$p \xrightarrow{a(t)} \top \iff a(r) \sqsubseteq p$$

$$p \xrightarrow{a(t)} \top \iff a(r) \cdot p \sqsubseteq p$$

$$p \xrightarrow{\delta(t)} \top \iff \delta(r) \sqsubseteq p$$
3. AN ABSTRACT OPERATIONAL SEMANTICS FOR BFAp6

Proof.
\[ \implies \text{By induction on the length of a derivation of the transition.} \]
\[ \iff \text{By the soundness of the axioms A1 and A2 and the definition of } \sqsubseteq. \]
\[ \square \]

For basic terms we have a Lemma which states that the (absolute) time shift operator does not filter anything away in the action rule for sequential composition, in other words, basic terms have ascending timestamps:

**Lemma 3.3.5** Basic terms have ascending timestamps
\[ \forall p \in B \quad p \xrightarrow{a(r)} r \gg p' \implies r \gg p' \iff p' \]

**Proof.** From the action rules we know that if \( q \xrightarrow{a(t)} q' \) with \( t \succ s \), then \( s \gg q \xrightarrow{a(t)} q' \) as well. By definition of basic terms and \( a(r) \cdot p' \subseteq p \) we know \( p' \in B(r) \). Hence, every transition of \( p' \) has a timestamp greater than \( r \). And thus, if \( p' \xrightarrow{b(t)} p'' \) then \( r \gg p' \xrightarrow{b(t)} p'' \) as well for all \( b \in A \). Similarly for terminating and deadlock transitions of \( p' \).

The consequence of this Lemma is, that for basic terms the application of the (absolute) time shift is not necessary anymore in the action rule for sequential composition, thus the action rule

\[
\frac{p \xrightarrow{a(r)} \checkmark}{p \cdot q \xrightarrow{a(r)} r \gg q} \quad \text{can be simplified to} \quad \frac{p \xrightarrow{a(r)} \checkmark}{p \cdot q \xrightarrow{a(r)} q}
\]

Hence, the action rules for the semantics of ACP ([Gla87]) are sufficient for basic terms.

**Lemma 3.3.6** The operational semantics of the previous section is equivalent with the one above, in the sense that for all \( p, q \in T \) and \( R \subseteq T \times T \):

\[ p \equiv q \quad \text{according to } R \quad \iff \quad p \equiv q \quad \text{according to } R \]

**Proof.**

- \( \implies \) By induction on the size of \( p \), so assume it has been proven for smaller states and consider \( (p, q) \in R \). Then there are two cases:
  - \( U(p) = L(p) \) then \( U'(p) = U(q) = L(q) = L(p) \) and neither \( p \) nor \( q \) has a deadlock. Since both definitions are equivalent for terms without deadlocks we may conclude \( p \equiv q \).
  - \( U(p) > L(p) \), but then also \( U(p) = U(q) > L(q) = L(p) \) and we have \( (u = U(p)) \)

\[
p \xrightarrow{\xi(u)} \checkmark \quad \text{and} \quad q \xrightarrow{\xi(u)} \checkmark
\]

Since both definitions are equivalent with respect to non-deadlock transitions and \( p \) and \( q \) have the same deadlock transition we may conclude \( p \equiv q \).

- \( \iff \) Analogously

\[ \square \]
3.3 An Abstract Operational Semantics with δ-Rules

Equivalence with the Original Operational Semantics

In this section we show that the abstract operational semantics without time component as presented in this report is equivalent with the original semantics from [BB89] as given in section 1. Remember that in the original semantics every state has both an expression component and a time-component. Consider

\[
\begin{align*}
\text{in abstract operational semantics} & \quad (a(r) \cdot p) \cdot q & \xrightarrow{a(r)} & (r \gg p) \cdot q \\
\text{in original operational semantics} & \quad < (a(r) \cdot p) \cdot q, 0 > & \xrightarrow{a(r)} & < p \cdot q, r >
\end{align*}
\]

We see that in the abstract semantics the course of time is encoded by an occurrence of \( r \gg \) somewhere in the prefix, while in the original semantics the course of time is being reflected by the time component in the state.

In order to relate the states of the (abstract) transition system of \( p \) with the states of the (original) transition \( < p, 0 > \), we define two functions inductively.

\[
\begin{align*}
\text{strip}(a(r)) & = a(r) & \text{time}(a(r)) & = 0 \\
\text{strip}(X + Y) & = X + Y & \text{time}(X + Y) & = 0 \\
\text{strip}(X \cdot Y) & = \text{strip}(X) \cdot Y & \text{time}(X \cdot Y) & = \text{time}(X) \\
\text{strip}(r \gg X) & = X & \text{time}(r \gg X) & = r
\end{align*}
\]

Now we can state some simple Lemma's concerning the (abstract) transition system \( \rightarrow \subseteq T \times T \):

Every resulting state contains an occurrence of \( r \gg \) in its prefix, which can be shown by the following Lemma:

**Lemma 3.3.7** \( \forall p, p' \in T \forall a \in A \forall r \in \mathbb{R}^0 \)

\[
p \xrightarrow{a(r)} p' \iff \text{time}(p') = r
\]

**Proof.** By induction on the length of the derivation. \( \square \)

Moreover, we have

**Lemma 3.3.8** \( \forall p, p' \in T \forall a \in A \forall r \in \mathbb{R}^0 \)

\[
p \xrightarrow{a(r)} p' \iff r \triangleright \text{time}(p) \land \text{strip}(p) \xrightarrow{a(r)} p'
\]

**Proof.** Both directions can be proven by induction on the length of the derivation. \( \square \)

The following Lemma states a correspondence between the transition system of \( p \) in the abstract operational semantics and the transition system of \( < \text{strip}(p), \text{time}(p) > \) in the original semantics:

**Lemma 3.3.9** \( \forall p, q \in T \forall a \in A \forall r \in \mathbb{R}^0 \)

\[
\begin{align*}
p \xrightarrow{a(r)} & \quad \iff \quad < \text{strip}(p), \text{time}(p) > \xrightarrow{a(r)} < \sqrt{r}, r > \\
p \xrightarrow{a(r)} q & \quad \iff \quad < \text{strip}(p), \text{time}(p) > \xrightarrow{a(r)} < \text{strip}(q), r > \\
< \text{strip}(p), \text{time}(p) > \xrightarrow{a(r)} < q, r > & \quad \iff \quad \exists q' \quad \text{strip}(q') \equiv q \land \text{time}(q') = r \land p \xrightarrow{a(r)} q'
\end{align*}
\]

**Proof.** All cases can be proven by induction on the length of the derivation. In some cases the previous Lemma must be used. \( \square \)

Furthermore, w.r.t. the original semantics we know that \( U(p) = \sup \{ s \mid < p, t > \rightarrow < p, s > \} \) and that \( U(p) > L(p) \) implies that \( < p, t > \xrightarrow{\delta(U(p))} \) for all \( b \) in \( A \).

Now we are ready to prove the following:
Theorem 3.3.10 The operational semantics presented in this section for BPA_6 is equivalent with the original semantics for ACP_6:
\[ \forall p, q \in T \quad p \leftrightarrow q \iff <p, 0> \leftrightarrow <q, 0> \]

Proof. First we prove by induction on \((\text{size}(p), \text{size}(q))\) that
\[ \forall p, q \in T \quad \text{time}(p) = \text{time}(q) : \quad p \leftrightarrow q \iff <\text{strip}(p), \text{time}(p)> \iff <\text{strip}(q), \text{time}(q)> . \]

Next, we have to show both directions of \(p \leftrightarrow q \iff <p, 0> \leftrightarrow <q, 0>\):

- \(\implies\) Assume \(R \subseteq T \times T\) is a bisimulation relation containing \((p, q)\). Note that for all \(p_0, q_0 \in T\)
  \((p_0, q_0) \in R\) implies \(U(p_0) = U(q_0)\). Construct \(R^* \subseteq (T \times \mathbb{R}^0) \times (T \times \mathbb{R}^0)\):
  \[ R^* = \{ (<\text{strip}(p_1), r>, <\text{strip}(q_1), r>) | (p_1, q_1) \in R \land 0 \leq r < U(p_1) \} \]
  \[ \cup \{ (<p, r>, <q, r>) | 0 \leq r < U(p) \} \]

The proof that \(R^*\) is a bisimulation relation is left to the reader.

- \(\iff\) Assume \(R^* \subseteq (T \times \mathbb{R}^0) \times (T \times \mathbb{R}^0)\) is a bisimulation relation containing \(<p, 0>, <q, 0>\),
  then construct \(R \subseteq T \times T\):
  \[ R = \{ (p_0, q_0) | (<\text{strip}(p_0), \text{time}(q_0)>, <\text{strip}(q_0), \text{time}(q_0)>) \in R^* \} \]
  \[ \cup \{ (p, q) \} \]

Again, the proof that \(R\) is a bisimulation relation is left to the reader.

\[ \square \]

3.4 An Abstract Operational Semantics for BPA_6 in \textit{n t y f t} format

In the previous section we introduced a model for BPA_6 by giving a Transition System Specification and we stated the soundness of the axioms of BPA_6. In order to obtain the soundness of the theory BPA_6 we need to prove as well that bisimulation equivalence is a congruence. We do this by using a result of [Gro89]. Groote proved in this paper that if the action rules of a Transition System Specification are in a specific format, which he called the \textit{n t y f t/ntyef}-format, then bisimulation equivalence is a congruence. In appendix B a short introduction to the ideas and results of [Gro89] is given. In this section we give a Transition System Specification in \textit{n t y f t}-format, which is an instantiation of the \textit{n t y f t/ntyef}-format. Since we prove as well that the induced bisimulation equivalence is equal to the one of the previous section we obtain the required congruence result.

A transition system specified by a Transition System Specification in \textit{n t y f t} format concerns only one relation. Therefore we cannot have a separate relation containing all the terminating transitions, instead we take the solution to encode termination in the label. (This idea can be found in [Bri88] and [BV89] as well)

We denote the transition relation, which is defined in this section, by \(\rightsquigarrow\) to distinguish it from the transition relation \(\rightarrow\) of the previous sections. We extend the set of labels with encodings of terminating transitions:

\[ A_\forall = \{ a, \forall a \in A \} \]
\[ A_\text{time} = \{ a(t) | a \in A \land t \in \mathbb{R}^0 \} \]
\[ \rightsquigarrow = T \times (A_\text{time} \cup A_\forall) \times T \]

Now we can encode in the label whether a transition is terminating or not. Instead of

\[ p \xrightarrow{a(r)} \sqrt{\text{which is an abbreviation of } (p, a(r)) \in \text{Terminate}} \]
we now have

\[ p \xrightarrow{a(r)} \delta \] which is an abbreviation of \((p, a(r)), \delta) \in \cdots\]

The next table, Table 6, contains a Transition System Specification for \(\text{BPA}_\delta\) in \(\text{ntyft}\) format.

\[
\begin{align*}
\text{atom} & : a(r) \xrightarrow{a(r)} \delta \\
\Rightarrow & : p \xrightarrow{a(r)} p' \\
& \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \q
Theorem 3.4.2 Bisimulation Equivalence is a congruence

Proof. We use a result of [Gro89]; if the Transition System Specification is well founded and the action rules are in a specific format, called the nstwft format, and the action rules can be stratified by a stratification function, then bisimulation equivalence is a congruence. A short treatment of this is given in Appendix B, however for a full understanding we refer to [Gro89]. Since the action rules of Table 6 are in the nstwft format it is sufficient to give a stratification. Instead of saying that we have one operator $\Rightarrow$ which takes a real number and a process term we can say also that for each $r \in \mathbb{R}^\geq 0$ there is an operator $\Rightarrow: T \rightarrow T$. Moreover $\delta$ can be seen as a constructor with rank zero. A stratification that does the job is the following:

$$\forall p, p', r, a \neq \delta \quad n(p \xrightarrow{a(r)} p') = 0$$
$$\forall p, p', r \quad n(p \xrightarrow{\delta(r)} p') = \text{size}(p)$$

Lemma 3.4.3 $\forall p \in T \forall u \in \mathbb{R}$

$$u = U(p) > L(p) \iff p \xrightarrow{\delta(r)} \delta$$

Proof. See Appendix C

The equivalence of the transition systems $\ldots$ and $\ldots$

First we state the following Lemma

Lemma 3.4.4 $\forall p, p' \in T \forall a \in A \forall r \in \mathbb{R}^\geq 0$

$$p \xrightarrow{a(r)} p' \iff p \xrightarrow{a(r)} p'$$
$$p \xrightarrow{a(r)} \top \iff p \xrightarrow{a(r)} \delta$$
$$p \xrightarrow{\delta(r)} \top \iff p \xrightarrow{\delta(r)} \delta$$

Proof. Left to the reader

From this Lemma we conclude

Theorem 3.4.5 $\forall p, q \in T$

$$p \equiv q \iff p \equiv q$$

Proof. From Lemma 3.4.4 it follows $R \subset T \times T$ containing $(p, q)$ is a bisimulation relation according to Definition 3.3.1 (for $\equiv$) if it is a bisimulation relation according to Definition 3.4.1 (for $\equiv$).
3.5 Completeness of BPA_{\delta}

Since we know that all the semantics which have been given so far, are equivalent with each other, it does not matter for which one we prove the completeness. We will continue with the semantics of the bisimulation equivalence \( \leftrightarrow \) associated with the Transition System Specification with \( \delta\)-rules of Table 5.

The theory BPA_{\delta} is complete w.r.t. the operational semantics if two arbitrary terms have bisimilar transition systems.

**Theorem 3.5.1 Completeness**

\[ \forall p,q \in T \quad p \leftrightarrow q \implies \text{BPA}_{\delta} \vdash p = q \]

It is sufficient to consider basic terms only.

**Proof.** First we will prove by induction on the size of \( p \) that \( p \subseteq q \). Assume

\[
\begin{align*}
    p &= \sum_i a_i(r_i) \cdot p_i + \sum_j b_j(s_j) \\
    q &= \sum_k c_k(t_k) \cdot q_k + \sum_l d_l(v_l)
\end{align*}
\]

\[
\forall i \quad p \xrightarrow{a_i(r_i)} r_i \Rightarrow p_i \quad \text{by bisimulation also } q \text{ has a transition} \\
\exists q' \quad q \xrightarrow{a_i(r_i)} r_i \Rightarrow q' \quad \text{with } r_i \Rightarrow p_i \leftrightarrow r_i \Rightarrow q' .
\]

Moreover, \( p \) and \( q \) are basic terms, hence by transitivity of \( \Rightarrow \) and by induction together with

\[
\begin{align*}
    p_i \equiv r_i \Rightarrow p_i \equiv r_i \Rightarrow q' \equiv q' \quad \text{we may conclude} \\
    a_i(i_i) \cdot q' \subseteq q \quad \text{and by transitivity of } \equiv \\
    a_i(i_i) \cdot p_i \subseteq q
\end{align*}
\]

\[
\forall j \quad p \xrightarrow{b_j(s_j)} q \quad \text{by bisimulation also } q \text{ has a transition} \\
q \xrightarrow{b_j(s_j)} q \quad \text{and thus}
\]

Adding these results together we get \( p \subseteq q \). Since bisimulation is symmetric we may conclude that \( q \subseteq p \) as well. From \( p \subseteq q \land q \subseteq p \) we conclude \( \text{BPA}_{\delta} \vdash p = q \).

4 Parallelism and Synchronisation

4.1 Introduction

Now we are ready to introduce parallelism and synchronisation resulting in the theory ACP_{\rho} from [BB83]. We will use as much as possible from ACP (without time) and we shall discuss only those cases in which we have to take the time information into account. One of the operators is the left-merge, which is an auxiliary operator that allows us to define the parallel merge operator \( \parallel \) in finitely many axioms. In standard ACP (without time) the term \( (a \cdot X) \parallel Y \) denotes the process in which the left component \( a \cdot X \) executes its first action \( a \), resulting in \( X \parallel Y \). In our real time setting it is a bit more subtle; if we take for example the process \( (a(t) \cdot X) \parallel Y \) then by executing the \( a \) action on time \( t \) the whole process must proceed in time. Whenever \( Y \) can wait after \( t \) (if \( t < \mathcal{U}(Y) \)) then \( (a(t) \cdot X) \parallel Y \).
can execute a action on time t, resulting in \((t \geq X)[(t \geq Y)]\), otherwise \((t \geq U(Y))\) a deadlock at time \(U(Y)\) occurs. Consider:

\[
\begin{align*}
a(2) \parallel b(3) &= a(2) \cdot b(3) \\
b(3) \parallel a(2) &= \delta(2)
\end{align*}
\]

In the first example the right component \(b(3)\) can wait until the left component \(a(2)\) executes its first action. In the second example however, we see that the right component \(a(2)\) cannot wait long enough and a deadlock at the ultimate delay of the right component is the result. The point is that in the process denoted by \(X \parallel Y\) only those initial actions of \(X\) can be executed, which have a timestamp smaller than \(U(Y)\). Hence, we need an auxiliary operator, called the bounded initialization. The bounded initialization has as input a process \(X\) and a time \(t\) and delivers that part of \(X\) which starts before \(t\). This new operator is the counterpart of the (absolute) time shift operator, which is denoted \(t \gg X\), as presented earlier. Therefore we shall denote the bounded initialization of \(X\) to \(t\) by \(X \gg t\).

**Definition 4.1.1** \(T\), with typical elements \(p, p_1, p_2\), is the set of terms over \(AC\p; p\), where \(a \in A\), \(r \in \mathbb{R}^{\geq 0}\) and \(H \subseteq A:\)

\[
p \in T \quad p := a(r) \mid p_1 + p_2 \mid p_1 \cdot p_2 \mid r \gg p \mid p \gg r \mid p_1 || p_2 \mid p_1 || p_2 \mid p_1 \parallel p_2 \mid \delta_H(p)
\]

We have to extend the definition of the syntactic operators for terms with parallelism, left merge and synchronization. \(i\)\(ntact(p)\) is an auxiliary operator, which delivers the set of initial timed actions which can be executed by \(p\). We need this operator since the the timestamps of the actions which can be executed by \(p|g\) are dependent on the timed actions of \(p\) and \(q\) and not only on the associated timestamps.

**Definition 4.1.2** Definition of \(U(p)\), \(i\)\(ntact(p)\), \(i\)\(ntim(p)\), \(L(p)\). For \(U(p)\) only the additional rules are given.

\[
a \in A
\]

\[
\begin{align*}
U(X \gg r) &= \min(U(X), r) \\
U(X||Y) &= \min(U(X), U(Y)) \\
U(X \parallel Y) &= \min(U(X), U(Y)) \\
U(X[Y]) &= \min(U(X), U(Y)) \\
U(\delta_H(X)) &= U(X)
\end{align*}
\]

\[
\begin{align*}
i\text{ntact}(\delta(t)) &= \emptyset \\
i\text{ntact}(a(t)) &= \begin{cases} \emptyset & \text{if } t = 0 \\ \{a(t)\} & \text{otherwise} \end{cases} \\
i\text{ntact}(X \cdot Y) &= \text{ntact}(X) \\
i\text{ntact}(X + Y) &= \text{ntact}(X) \cup \text{ntact}(Y) \\
i\text{ntact}(r \gg X) &= \{a(t) \mid a(t) \in \text{ntact}(X) \land t > r\} \\
i\text{ntact}(X \gg r) &= \{a(t) \mid a(t) \in \text{ntact}(X) \land t < r\} \\
i\text{ntact}(X|Y) &= \{c[t] \mid \exists a, b \land a(t) \in \text{ntact}(X) \land b(t) \in \text{ntact}(Y) \land a|b = c \neq \delta\} \\
i\text{ntact}(X \parallel Y) &= \{a(t) \mid a(t) \in \text{ntact}(X) \land t < U(Y)\} \\
i\text{ntact}(X \parallel Y) &= \text{ntact}(X|Y) \cup \text{ntact}(X \parallel Y) \cup \text{ntact}(Y \parallel X) \\
i\text{ntact}(\delta_H(X)) &= \{a(t) \mid a(t) \in \text{ntact}(X) \land a \neq H\}
\end{align*}
\]

\[
\begin{align*}
i\text{ntim}(X) &= \{t \mid \exists a \in A : a(t) \in \text{ntact}(p)\} \\
L(X) &= \max(i\text{ntim}(X)) \text{ where } \max(\emptyset) = 0
\end{align*}
\]

We do not extend the definitions of \(\leq\) and \(\subseteq\), hence these relations are defined for terms over \(BPA\phi\) only. We don't need these relations in the sequel.
4.2 The Theory ACPρ

We assume a communication function \( | : A_δ \times A_δ \rightarrow A_δ \) which is symmetric, commutative and associative and has \( δ \) as zero element. We don’t need time information in the definition of this communication function because in our view it does not make sense to define a communication between atomic actions which happen at a different time. Thus if \( \bar{a}(a = ε) \) then

\[
\bar{a}(2)|a(2) = c(2)\\
\bar{a}(1)|a(3) = δ(1)
\]

### 4.2 The Theory ACPρ

The axiom system for ACPρ is given in the Table 7, some remarks on these axioms are given below.

<table>
<thead>
<tr>
<th>( a, b \in A_δ )</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATB5</td>
</tr>
<tr>
<td>ATB6</td>
</tr>
<tr>
<td>ATB7</td>
</tr>
<tr>
<td>ATB8</td>
</tr>
<tr>
<td>ATC1</td>
</tr>
<tr>
<td>ATC2</td>
</tr>
<tr>
<td>CM1</td>
</tr>
<tr>
<td>ATCM2</td>
</tr>
<tr>
<td>ATCM3</td>
</tr>
<tr>
<td>CM4</td>
</tr>
<tr>
<td>CM5</td>
</tr>
<tr>
<td>CM6</td>
</tr>
<tr>
<td>CM7</td>
</tr>
<tr>
<td>CM8</td>
</tr>
<tr>
<td>CM9</td>
</tr>
</tbody>
</table>

| D1 | \( a \notin H \) | \( δ_H(a) = ε \) |
| D2 | \( a \in H \) | \( δ_H(a) = δ \) |
| ATD | \( \delta_H(a(t)) = (\delta_H(a))(t) \) |
| D3 | \( δ_H(X + Y) = δ_H(X) + δ_H(Y) \) |
| D4 | \( δ_H(X \cdot Y) = δ_H(X) \cdot δ_H(Y) \) |

**Table 7**: ACPρ = BPAρδ + ATB5-8 + ATC1,2 + CM1 + ATCM2,3 + CM4-9 + D1-4 + ATD

**ATC1** Both processes have to wait in parallel. Synchronisation is not possible when the time stamps are not equal and a deadlock occurs at the moment that one of the initial actions is not active any longer.

**ATC2** If the time stamps are equal then synchronisation may occur according to the communication function.
CM1 This axiom is exactly the same as in ACP, however together with the axioms for the left merge it does not result in arbitrary interleaving. Now the time stamps of the atomic actions determine the possible orderings.
\[
a(2) \parallel b(3) = a(2) \parallel b(3) + b(3) \parallel a(2) + a(2) \parallel b(3) = a(2) \cdot b(3) + \delta(2) + \delta(2) = a(2) \cdot b(3)
\]

ATCM2,3 In a left merge we have to ensure that the right component can wait long enough, otherwise the left merge delivers a deadlock at the moment that the right component cannot wait any longer.
\[
a(2) \parallel b(3) = (a(2) \gg 3) \cdot b(3) = c(2) \cdot b(3)
b(3) \parallel a(2) = (b(3) \gg 2) \cdot a(2) = \ell(2) \cdot a(2) = \ell(2)
\]

Note that the definition of the ultimate delay operator is extended to all operators, which is not the case in [BB89]. Moreover, we can check that the definitions of all the syntactic operators are consistent with the theory, which means for example that \(ACP \vdash X = Y \implies U(X) = U(Y)\)

**Lemma 4.2.1** For each \(p, q \in B\) there is a \(z \in B\) such that \(ACP \vdash z = p \parallel q\)

**Proof.** Assume
\[
\begin{align*}
p & \approx \sum_{i \in I} a_i(r_i) \cdot p_i + \sum_{j \in J} b_j(s_j) \\
q & \approx \sum_{k \in K} c_k(t_k) \cdot q_k + \sum_{i \in L} d_i(u_i)
\end{align*}
\]

We proof the Lemma by induction on the size of \(p\). We give only the atomic case, the other case can be done analogously. Define the following subsets
\[
\begin{align*}
J^t & = \{ j \in J \mid s_j > t \} \\
I^t & = \{ i \in I \mid u_i > t \} \\
J_L & = \{ j \in J \mid L^{s_j} \neq \emptyset \} \\
I_L & = \{ i \in I \mid J^{u_i} \neq \emptyset \}
\end{align*}
\]

We can construct a \(z \in B\) such that \(z = p \parallel q\) by taking
\[
z \approx \sum_{j \in J_L} b_j(s_j) \cdot \sum_{l \in L^{s_j}} d_l(u_l) + \sum_{i \in I_L} d_i(u_i) \cdot \sum_{j \in J^{u_i}} b_j(s_j) + \sum_{j \in J \times L, s_j = u_i} (b_j \cdot d_l)(s_j)
\]

**Lemma 4.2.2** For each \(p \in B\) there is a \(z \in B\) such that \(ACP \vdash z = \delta_R(p)\)

**Proof.** By simple induction on the size of \(p\).

**Theorem 4.2.3** Elimination Theorem

For each term \(p \in T\) there is a basic term \(p_b\) such that \(ACP \vdash p = p_b\)

**Proof.** The proof uses induction on the size of \(p\). We have to discuss four cases, the other cases can be proved as in Lemma 2.2.4:
1. \( p \equiv q \mid r \) By induction there are \( q_b, r_b \in B \) such that \( ACP \rho \vdash q = q_b \) and \( ACP \rho \vdash r = r_b \) with

\[
q_b \simeq \sum_{i \in I} a_i(r_i) \cdot q_i + \sum_{j \in J} b_j(s_j) \\
r_b \simeq \sum_{k \in K} c_k(t_k) \cdot r_k + \sum_{l \in L} d_l(u_l)
\]

After applying CM8 and CM9 sufficiently many times we get

\[
p = \sum_{(i, k) \in I \times K} a_i(r_i) \cdot c_k(t_k) \cdot r_k + \sum_{(i, l) \in I \times L} a_i(r_i) \cdot d_l(u_l)
\]

Each \( a(t)[k(t') \vdash \) reduces to some \( c(t) \) or \( \delta(min(t, t')) \) and we know that a term not containing \( ||, \) \( \llbracket \) or \( | \) has a basic term. By the previous Lemma 4.2.1 there is for each \( (i, k) \in I \times K \) a \( p(i, j) \in B \) such that \( ACP \rho \vdash p(i, j) = q_i \mid r_k \). Construct the following subsets of index pairs:

\[
I \times K = \{ (i, k) \in I \times K \mid a_i \neq \delta \wedge r_i = t_k \} \\
I \times L = \{ (i, l) \in I \times L \mid a_i \neq \delta \wedge r_i = u_l \} \\
J \times K = \{ (j, k) \in J \times K \mid b_j \neq \delta \wedge s_j = t_k \} \\
J \times L = \{ (j, l) \in J \times L \mid b_j \neq \delta \wedge s_j = u_l \}
\]

Next, construct \( \bar{p} \):

\[
\bar{p} \simeq \sum_{(i, k) \in I \times K} a_i(r_i) \cdot c_k(t_k) \cdot r_k + \sum_{(i, l) \in I \times L} a_i(d_l(u_l)) \cdot q_i + \sum_{(j, k) \in J \times K} b_j(s_j) \cdot c_k(t_k) \cdot r_k + \sum_{(j, l) \in J \times L} b_j(d_l(u_l)) \cdot q_i
\]

And we can conclude

\[
p = \begin{cases} \\
\bar{p} + \delta(U(p)) \in B & \text{if } U(p) > U(\bar{p}) \\
\bar{p} \in B & \text{otherwise}
\end{cases}
\]

2. \( p \equiv q \| r \) By induction there are \( q_b, r_b \in B \) such that \( ACP \rho \vdash q = q_b \) and \( ACP \rho \vdash r = r_b \) with

\[
q_b \simeq \sum_{i \in I} a_i(r_i) \cdot q_i + \sum_{j \in J} b_j(s_j) \\
r_b \simeq \sum_{k \in K} c_k(t_k) \cdot r_k + \sum_{l \in L} d_l(u_l)
\]

Define for arbitrary \( t \in R^{\geq 0} \)

\[
I^t = \{ i : i \in I \wedge t_i < t \} \quad \text{and} \quad J^t = \{ j : j \in J \wedge s_j < t \}
\]

After applying CM4 and ATCM2,3 sufficiently many times we get

\[
p = \sum_{i \in I} (a_i(r_i) \gg U(r_b)) \cdot q_i + \sum_{j \in J} (b_j(s_j) \gg U(r_b)) \cdot r_b
\]

Take \( U(r_b) = u \). By the previous Lemma 4.2.1 there is for each \( i \in I^u \) a \( p_i \in B \) such that \( p_i = q_i \| r_b \) and we can construct \( \bar{p} \) (note \( \forall i \in I^u : a_i(r_i) \gg u = a_i(r_i) \))

\[
\bar{p} \simeq \sum_{i \in I^u} a_i(r_i) \cdot p_i + \sum_{j \in J^u} b_j(s_j) \cdot r_b
\]

And we can conclude

\[
p = \begin{cases} \\
\bar{p} \in B & \text{if } I^u \cup J^u = \emptyset \\
\bar{p} + \delta(u) \in B & \text{if } I^u \cup J^u \neq \emptyset \wedge (I - I^u) \cup (J - J^u) = \emptyset
\end{cases}
\]
3. \( p \equiv q \| r \) This follows from CM1 and the first two cases

4. \( p \equiv \delta_H(q) \) By induction there is a \( q_b \in E \) such that \( ACPp \vdash q = q_b \), say

\[
q_b \simeq \sum_{i \in I} a_i(r_i) \cdot q_i + \sum_{j \in J} b_j(s_j)
\]

And the case follows by the previous Lemma 4.2.2.

\[\square\]

### 4.3 An Abstract Operational Semantics for ACP\( p \) with \( \delta \) transitions

We add some rules to the operational semantics of BPA\( \delta \) for the new operators \( \|, \| \) and \( \ll \). Remember that if the process \( a(r) \cdot p \) executes the \( a(r) \) action, it evolves to \( r \gg p \) denoting that the time has increased to \( r \) and that only the part of \( p \) starting after \( r \) may continue.

If \( a(r) \cdot p \) executes this \( a(r) \) in a parallel composition with \( q \), the "increase its time" must hold for \( q \) as well. In other words, \( (a(r) \cdot p)\| q \) evolves to \( (r \gg p)\| (r \gg q) \) by executing \( a \) at time \( r \) whenever \( q \) can wait until \( r \). The "check" whether \( q \) can increase its time, is denoted by \( r < U(q) \). Hence, we obtain the following action rule:

\[
\frac{p \xrightarrow{a(r)} p' \quad r < U(q)}{p\| q \xrightarrow{a(r)} p'\| (r \gg q) \quad e\| p \xrightarrow{a(c)} (r \gg q)\| p'}
\]

Note that if \( p \xrightarrow{a(r)} p' \) then \( p' \) contains one or more occurrences of \( r \gg \) (e.g. \( \text{time}(p') = r \)); hence, only transitions with a timestamp greater than \( r \) are allowed for \( p' \). In the sequel we will extend the definitions of \( \text{strip}() \) and \( \text{time}() \) to terms with parallelism.

If there is a synchronisation then both components have to execute an action:

\[
\frac{p \xrightarrow{a(r)} p' \quad q \xrightarrow{b(r)} q' \quad a|b = c \neq \delta}{p\| q \xrightarrow{a(r)} p'\| q'}
\]

The rules for the bounded initialization are symmetric to the action rules for the (absolute) time shift as given in Table 4. The Transition System Specification of the operational semantics of ACP\( p \) is given in Table 8. Bisimulation equivalence defined in Definition 3.3.1 is denoted by \( \equiv \).

**Theorem 4.3.1** The axioms of ACP\( p \) are sound with respect to bisimulation equivalence.

**Proof.** Straightforward

\[\square\]

### Equivalence with the Original Operational Semantics

We extend the definition \( \text{strip} \) and \( \text{time} \) (see section 3.3):

- \( \text{strip}(X\|Y) = \text{strip}(X)\|\text{strip}(Y) \)
- \( \text{time}(X\|Y) = \begin{cases} \text{time}(X) & \text{if } \text{time}(X) = \text{time}(Y) \\ \text{undefined} & \text{otherwise} \end{cases} \)
- \( \text{strip}(X\ll Y) = X\ll Y \)
- \( \text{time}(X\ll Y) = 0 \)
- \( \text{strip}(X\|Y) = X\|Y \)
- \( \text{time}(X\|Y) = 0 \)
- \( \text{strip}(r \gg X) = X \)
- \( \text{time}(r \gg X) = r \)

Now we need a Lemma which states that \( \text{time}() \) is always defined in a resulting state:
4.3 An Abstract Operational Semantics for ACP\(\rho\) with \(\delta\) transitions

<table>
<thead>
<tr>
<th>Import rules from Table 4</th>
</tr>
</thead>
</table>

\(a \in A\)

\[
\frac{p \xrightarrow{a(r)} p'}{p || q \xrightarrow{a(r)} p'||(r \geq q)}, \quad q || p \xrightarrow{a(r)} (r \geq q)||p', \quad p \parallel q \xrightarrow{a(r)} p'||(r \geq q)
\]

\[
\frac{p \xrightarrow{a(r)} \sqrt{r < U(q)}}{p || q \xrightarrow{a(r)} r \geq q, \quad q || p \xrightarrow{a(r)} r \geq q, \quad p \parallel q \xrightarrow{a(r)} r \geq q}
\]

\[
\frac{p \xrightarrow{a(r)} p', \quad q \xrightarrow{b(r)} q', \quad a|b = c \neq \varepsilon}{p || q \xrightarrow{c(r)} p'||q'}, \quad p || q \xrightarrow{c(r)} p'||q', \quad p \parallel q \xrightarrow{c(r)} \sqrt{r}, \quad p || q \xrightarrow{c(r)} \sqrt{r}
\]

\[
\frac{p \xrightarrow{a(r)} \sqrt{r \geq q}, \quad q || p \xrightarrow{c(r)} q', \quad p || q \xrightarrow{c(r)} q', \quad p \parallel q \xrightarrow{c(r)} q'}{p || q \xrightarrow{c(r)} q', \quad q || p \xrightarrow{c(r)} q', \quad p || q \xrightarrow{c(r)} q', \quad q || p \xrightarrow{c(r)} q'}
\]

\[
\frac{s \geq r \quad p \xrightarrow{a(r)} p'}{p \gg s \xrightarrow{a(r)} p'}
\]

\[
\frac{p \xrightarrow{a(r)} \lambda \notin H}{\delta_H(p) \xrightarrow{a(r)} \delta_F(p')}
\]

\[
\frac{U(p) > L(p)}{p \xrightarrow{U(p)} \sqrt{r}}
\]

Table 8: Transition System Specification for ACP\(\rho\)
Lemma 4.3.2 \( \forall p, p_0, p_1 \in \mathcal{T} \forall v \in A \forall r \in \mathbb{R}^2 \rho \)

\[
p \xrightarrow{a(r)} p_0 || p_1 \quad \Longrightarrow \quad \text{time}(p_0) = \text{time}(p_1) = r
\]

Proof. We know already that if \( p \) is a term over BPA\( \rho \) then:

\[
p \xrightarrow{a(r)} p' \quad \Longrightarrow \quad \text{time}(p') = r
\]

and the Lemma follows direct by induction on the length of the derivation of \( p \xrightarrow{a(r)} q_0 || q_1 \). \( \qed \)

Since the Lemma's 3.3.7, 3.3.8 and 3.3.9 hold for terms over ACP\( \rho \) as well, we may conclude that the equivalence between \( \longrightarrow \) and \( \longleftrightarrow \), as stated in Theorem 3.3.10, does still hold.

4.4 An Abstract Operational Semantics for ACP\( \rho \) in nt\( yf \)t format

In this section we follow the idea of section 3.4 by changing in every premise of the action rules of Table 8 all occurrences of \( p \xrightarrow{a(c)} q \) into \( p \xrightarrow{a(r)} q \). The action rules for transitions labelled with timed actions from \( A^{times} \) are given in Table 9. A more difficult task is to give \( \delta \)-rules in nt\( yf \)t format. These action rules are given in Table 10. In this Table we use a unary function \( \circ \) on actions. If \( b \in A_\delta \) then \( \tilde{b} = b \) and if \( b = a, \circ \) for some \( a \in A \) then \( \tilde{b} = a \). At first sight the action rules of Table 10 look rather complicated. But note that

\[
\frac{U(p||q) > L(p||q)}{p||q \xrightarrow{\delta(U(p||q))} \delta}
\]

corresponds with the two rules

\[
\begin{align*}
\{ & u = U(p) \\
    u &= U(q) \\ 
    u &\notin \text{splittim}(p||q) \} \quad , \quad \{ & u = U(p) > L(p) \\
    u &\leq U(q) \} \\
\frac{p||q \xrightarrow{\delta(u)} \delta}{p||q \xrightarrow{\delta(u)} \delta} , & \quad \frac{q||p \xrightarrow{\delta(u)} \delta}{q||p \xrightarrow{\delta(u)} \delta}
\end{align*}
\]

And these last two rules correspond exactly with the \( \delta \) rules for \( || \) in Table 10.

**Theorem 4.4.1** Bisimulation Equivalence is a congruence

Proof. By the Theorem 3.4.2, since the stratification which is given in the proof of Theorem 3.4.2 is still valid. \( \qed \)

**Lemma 4.4.2** \( \forall p \in \mathcal{T} \forall u \in \mathbb{R}^2 \rho \)

\[
u = U(p) > L(p) \quad \iff \quad p \xrightarrow{\delta(u)} \delta
\]

Proof. See Appendix C. \( \qed \)

**Theorem 4.4.3** The two Transition System Specifications and the associated definitions of bisimulation induce the same equivalence.

\( \forall p, q \in \mathcal{T} \)

\[
p \leftrightarrow q \iff p \xrightarrow{\leftrightarrow} q
\]

Proof. Follows directly from the fact that 3.4.4 holds for terms over ACP\( \rho \) as well. \( \qed \)
4.4 An Abstract Operational Semantics for ACP\textsubscript{p} in ntuft format

<table>
<thead>
<tr>
<th>Import rules from Table 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a, a' \in A$, $b \in \mathcal{A}_b \cup \mathcal{A}_e$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\ll, \ll$</th>
<th>$p \xrightarrow{a(r)} p' \quad t &gt; r \quad q \xrightarrow{b(t)} q'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p \parallel q \xrightarrow{a(r)} p', q \parallel q\parallel (r \gg q)$, $q\parallel p \xrightarrow{a(r)} (r \gg q) \parallel p'$, $p \parallel q \xrightarrow{a(r)} p' \parallel (r \gg q)$</td>
<td></td>
</tr>
<tr>
<td>$p \xrightarrow{a\lor(r)} p' \quad t &gt; r \quad q \xrightarrow{b(t)} q'$</td>
<td></td>
</tr>
<tr>
<td>$p \parallel q \xrightarrow{a(r)} r \gg q$, $q\parallel p \xrightarrow{a(r)} r \gg q$, $p \parallel q \xrightarrow{a(r)} r \gg q$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\ll, \ll$</th>
<th>$p \xrightarrow{a(r)} p' \quad q \xrightarrow{a(r)} q' \quad a\parallel a' = c \neq \delta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p \parallel q \xrightarrow{a(r)} p' \parallel q'$, $p\parallel q \xrightarrow{a(r)} p' \parallel q'$</td>
<td></td>
</tr>
<tr>
<td>$p \parallel q \xrightarrow{a(r)} q', q\parallel p \xrightarrow{a(r)} q'$, $q\parallel p \xrightarrow{a(r)} q'$</td>
<td></td>
</tr>
<tr>
<td>$p \xrightarrow{a\lor(r)} p' \quad q \xrightarrow{a\lor(r)} q' \quad a\parallel a' = c \neq \delta$</td>
<td></td>
</tr>
<tr>
<td>$p \parallel q \xrightarrow{a(r)} q'$, $q\parallel p \xrightarrow{a(r)} q'$, $q\parallel p \xrightarrow{a(r)} q'$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\Rightarrow$</th>
<th>$s &gt; r \quad p \xrightarrow{b(r)} p'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p \Rightarrow s \xrightarrow{b(r)} p'$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\delta_H$</th>
<th>$p \xrightarrow{a(r)} p' \quad a \notin H$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta_H(p) \xrightarrow{a(r)} \delta_H(p')$</td>
<td></td>
</tr>
</tbody>
</table>

| $\delta_H(p) \xrightarrow{a\lor(r)} p'$ |

Table 9: First Part of Transition System Specification for ACP\textsubscript{p}
Table 10: Second Part of Transition System Specification for ACP in ntyft format
4.5 Completeness of ACP\(\rho\)

**Theorem 4.5.1** Completeness for ACP\(\rho\)

\[ p \leftrightarrow q \implies ACP\rho \vdash p = q \]

**Proof.** By the soundness of the axioms of ACP\(\rho\), Theorem 4.3.1, the Congruence Theorem 4.4.1 and the Elimination Theorem 4.2.3 we may conclude:

\[ \exists p_b, q_b \in B \quad p_b \Rightarrow p \Leftrightarrow q \Leftrightarrow q_b \]

By transitivity of bisimulation and the Completeness Theorem 3.5.1 we get

\[ ACP\rho; \vdash p = q \]

\[ \square \]

5 Integrals

5.1 Introduction

We call integration the alternative composition over a continuum of alternatives [BB89].

So, if an action \(a\) can happen somewhere in the interval \([1,2]\) we write:

\[ \int_{v \in [1,2]} a(v) \]

The bounds of an interval may be dependent on *time variables* which are defined in the context.

\[ \int_{v \in [1,2]} a(v) \cdot \int_{w \in [v+1,v+2]} b(w) \]

In this section we take a more restrictive view on integration than in [BB89], called *prefixed* integration; we require that every action has as timestamp a *time variable*, directly preceded by the binding integral. Hence, we do not consider "general" integration, integrals over arbitrary (open) terms, as in [BB86]. E.g. we allow the following terms:

\[ \int_{v \in V} a(v) \cdot \int_{w \in W} b(w) \quad \text{and} \quad \left( \int_{v \in V} a(v) + \int_{w \in W} b(w) \right) \cdot \int_{z \in Z} c(z) \]

but not

\[ \int_{v \in V} \left( \int_{w \in W} a(w) \right) \quad \text{or} \quad \int_{v \in V} \left( a(2) \cdot b(v) \right) \]

To prove important facts (like completeness) in detail, we have to introduce some definitions concerning bounds and intervals.
5.2 Some Definitions

- TVar is an infinite, countable set of time variables. The set of bounds, Bound, with typical elements \( v, v_1, v_2 \), is defined as follows, where \( t \in \mathbb{R}^{\geq 0}, \omega \in TVar \):

\[
v \in \text{Bound} \quad v := t \mid v_1 + v_2 \mid v_1 - v_2 \mid t \cdot v
\]

If \( v \in \text{Bound} \) then the set of time variables occurring in \( v \) is denoted by tvar(\( v \)). We take the freedom to assume the standard notion of identity and the standard ordering on Bound, e.g., \( 1 + 2 = 3, v - v = 0, v + 2 < v + 3 \) but \( v + 2 \not< w + 3 \). Moreover, we assume \( 0 < v \) and \( v < \omega \) for all \( v \in TVar \). Hence, the ordering on Bound is not total.

- The set of intervals \( \text{Int} \):

\[
\text{Int} = \text{Bool} \times \text{Bound} \times \text{Bound} \times \text{Bool}
\]

We abbreviate a set \( \{V_1, \ldots, V_n\} \) of intervals by \( \vec{V} \). In the sequel \( \langle, \rangle \) are variables ranging over Bool. An arbitrary interval \( \langle t, v_0, v_1, \rangle \) is abbreviated by \( \langle v_0, v_1 \rangle \). Furthermore, we abbreviate

\[
V_0 = (tt, v_0, v_1) \quad \text{by} \quad V_0 = [v_0, v_1]
\]

\[
V_1 = (ff, v_0, v_1) \quad \text{by} \quad V_1 = < v_0, v_1>
\]

denoting that \( V_0 \) has a closed lower bound and \( V_1 \) has an open lower bound. Similarly we abbreviate

\[
W_0 = (t, v_0, v_1, tt) \quad \text{by} \quad W_0 = \langle v_0, v_1 \rangle
\]

\[
W_1 = (t, v_0, v_1, ff) \quad \text{by} \quad W_1 = \langle v_0, v_1 \rangle
\]

denoting that \( W_0 \) has a closed upper bound and \( W_1 \) has an open upper bound. The set of time variables occurring in the Bound of an interval \( V \) is denoted by tvar(\( V \)). Intervals in which no time variables occur are called time-closed intervals.

\[
\text{Int}_{\text{time-closed}} = \text{Bool} \times \mathbb{R}^{\geq 0} \times \mathbb{R}^{\geq 0} \times \text{Bool}
\]

We use the following abbreviation for \( t \in \mathbb{R}^{\geq 0} \) and \( V \in \text{Int} \):

- If \( V = < v_0, v_1 > \) then \( t \in V \) abbreviates \( v_0 < t < v_1 \)
- If \( V = [v_0, v_1] \) then \( t \in V \) abbreviates \( v_0 \leq t \leq v_1 \)
- If \( V = < v_0, v_1 ] \) then \( t \in V \) abbreviates \( v_0 < t \leq v_1 \)
- If \( V = [v_0, v_1] \) then \( t \in V \) abbreviates \( v_0 \leq t \leq v_1 \)

Similarly, \( t \not\in V \) and \( V = \emptyset \) can be defined. Of course we have two (infix) operators \( \cap, \cup \in \text{Int}_{\text{time-closed}} \times \text{Int}_{\text{time-closed}} \rightarrow \text{Int}_{\text{time-closed}} \). The definitions of \( \cap \) and \( \cup \) are straightforward from the above definition of \( t \in V \), with the remark that \( V_1 \cup V_2 \) is defined if \( V_1 \cup V_2 \) is again an interval. Furthermore, for \( t \in \mathbb{R}^{\geq 0} \) \( V \in \text{Int} \) we write \( t < V \) if \( t \) is smaller than every element of \( V \).

- If \( V = < v_0, v_1 > \) then \( t < V \) abbreviates \( t < v_0 \)
- If \( V = [v_0, v_1] \) then \( t < V \) abbreviates \( t < v_0 \)

On \( \text{Int} \) two operators \( \sup, \inf \in \text{Int} \rightarrow \text{Bound} \) are defined:

Let \( V = \langle v_0, v_1 \rangle \) \& \( v_0, v_1 \in \text{Bound} \)

\[
\begin{align*}
V \neq \emptyset : & \quad \inf(V) = v_0 \\
& \quad \sup(V) = v_1 \\
V = \emptyset : & \quad \inf(V) = \sup(V) = 0
\end{align*}
\]

Note that all these expressions concerning intervals are in fact abbreviations of a conjunction of linear equations and linear inequalities over time variables and numbers from \( \mathbb{R}^{\geq 0} \).
5.2 Some Definitions

- In this section the set of terms over BPAδ, $T$, is defined inductively.
  Let $a \in A_δ$, $V \in \text{Int}$

$$\int_{v \in V} (a(v)) \in T$$

$\quad \Rightarrow \quad \int_{v \in V} (a(v) \cdot p) \in T$

$\quad p, q \in T \quad \Rightarrow \quad p \cdot q \in T$

$\quad p, q \in T \quad \Rightarrow \quad p + q \in T$

$\quad p \in T, u \in \text{Bound} \quad \Rightarrow \quad u \gg p \in T$

The time variable bound by a first integral of a term $p$ is called a leading time variable of $p$.

- We use the following abbreviation, for $w \in \text{Bound}$ and $a \in A_δ$:

$$\int_{v \in [w, w]} a(v) = a(w)$$

and again, $δ(0)$ is abbreviated by $δ$.

- $FV(p)$ is the set of free time variables in $p$.
  Let $p, q \in T$, $a \in A_δ$, $V \in \text{Int}$, $u \in \text{Bound}$ and $v \in TVar$

$$\begin{align*}
FV(\int_{v \in V} (a(v))) &= tvar(V) \\
FV(\int_{v \in V} (a(v) \cdot p)) &= (FV(p) - \{v\}) \cup tvar(V) \\
FV(p \cdot q) &= FV(p) \cup FV(q) \\
FV(p + q) &= FV(p) \cup FV(q) \\
FV(u \gg p) &= FV(p) \cup tvar(u)
\end{align*}$$

In the sequel we will use a scope convention, saying that we will not write the scope brackets if the scope is as large as possible. Thus we write $\int_{v \in V} a(v) \cdot p$ for $\int_{v \in V} (a(v) \cdot p)$ if we do not consider this term as the initial part of a sequential composition.

- A term $p$ with $FV(p) = \emptyset$ is called a time-closed term.

$$\mathcal{T}^{\text{time-closed}} = \{ p \in T | FV(p) = \emptyset \}$$

- We define a notion of substitution of free time variables. If $V \in \text{Int}$ then $V[w, v] \in \text{Int}$ denotes the interval in which all occurrences of the time variable $v$ in the bounds of $V$ are replaced by $w$. Similarly, we have a substitution on Boundary, like $u[w, v]$.

Let $p, q \in T$, $V \in \text{Int}$, $a \in A_δ$, $u \in \text{Bound}$, $v, v', v'' \in TVar$ where $v \neq v'$ and finally $w \in R^0 \cup TVar$.

$$\begin{align*}
(\int_{v' \in V} a(v''))[w/v] &= \int_{v'' \in V} a(v'') \\
(\int_{v' \in V} (a(v') \cdot p))[w/v] &= \int_{v'' \in V} (a(v') \cdot p) \\
(\int_{v' \in V} (a(v) \cdot p))[w/v] &= \int_{v'' \in V} a(v'') \cdot (p[w/v]) \\
(p + q)[w/v] &= p[w/v] + q[w/v] \\
(p \cdot q)[w/v] &= p[w/v] \cdot q[w/v] \\
(u \gg p)[w/v] &= u[w/v] \gg p[w/v]
\end{align*}$$

- From now on $≡$ denotes syntactic equivalence modulo $\alpha$-conversion.

Let $a \in A_δ$ and $v, w \in TVar$ where $w \notin FV(p)$

$$\begin{align*}
\int_{v \in V} a(v) &= \int_{v \in V} a(w) \\
\int_{v \in V} (a(v) \cdot p) &= \int_{w \in V} (a(w) \cdot p[w/v])
\end{align*}$$
5 INTEGRALS

- We have to adapt the definition of active(), initimm() etc. We give here only the action rules for the atomic cases (including $\int_{v \in V} a(v) \cdot p$). The other cases do not change.

Let $p \in T$, $a \in A$, $V \in Int_{time-closed}$ and $v \in TVar$

$$U(\int_{v \in V} \delta(v)) = U(\int_{v \in V} a(v)) = \sup(V)$$

$$initimm(\int_{v \in V} a(v)) = initimm(\int_{v \in V} (a(v) \cdot p)) = V$$

$$L(X) = \sup(initimm(X))$$

- Since we want to consider $\int_{v \in V'} a(v)$ as a syntactic summand of $\int_{v \in V} a(v)$ whenever $V' \subseteq V$, we have to add some rules to our definition of $\sqsubseteq$. Let $p, q \in T$, $V, V' \in Int_{time-closed}$, $a \in A$ and $v \in TVar$

$$V' \subseteq V \land \int_{v \in V'} a(v) \subseteq p \implies \int_{v \in V} a(v) \subseteq p$$

$$V' \subseteq V \land \int_{v \in V} a(v) \cdot q \subseteq p \implies \int_{v \in V} (a(v) \cdot q) \subseteq p$$

5.3 The Theory of BPAr6I

In this section we give an axiom system for BPAr6I which can be seen as an extension to and an adaptation of BPAr6 (as given in Table 3). We have four new axioms called INT1-4 which express the specific properties of terms with integration. The other axioms are the BPAr6I versions of BPAr6 axioms.

The axiom system for BPAr6I is given in Table 11

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT1</td>
<td>$V = V_1 \cup V_2$ $\int_{v \in V_1} a(v) + \int_{v \in V_2} a(v) = \int_{v \in V} a(v)$</td>
</tr>
<tr>
<td>INT2</td>
<td>$\int_{v \in \emptyset} a(v) = \emptyset$</td>
</tr>
<tr>
<td>INT3a</td>
<td>$v \not\in FV(Y)$ $\int_{v \in V} (a(v) \cdot Y) = \int_{v \in V} (a(v) \cdot (X \cdot Y))$</td>
</tr>
<tr>
<td>INT3b</td>
<td>$v \not\in FV(Y)$ $\int_{v \in V} (a(v) \cdot Y) = \int_{v \in V} (a(v) \cdot (X \cdot Y))$</td>
</tr>
<tr>
<td>INT4a</td>
<td>$\forall t \in V : X = X + a(v)[t/v] \implies X = X + \int_{v \in V} a(v)$</td>
</tr>
<tr>
<td>INT4b</td>
<td>$\forall t \in V : X = X + (a'(v) \cdot Y)[t/v] \implies X = X + \int_{v \in V} (a'(v) \cdot Y)$</td>
</tr>
<tr>
<td>INT5</td>
<td>$\forall t \in V : X[t/v] = Y[i/v] \implies \int_{v \in V} (a(v) \cdot X) = \int_{v \in V} (a(v) \cdot Y)$</td>
</tr>
<tr>
<td>AT1</td>
<td>$a(0) = \emptyset$</td>
</tr>
<tr>
<td>AT2</td>
<td>$\int_{v \in V} \delta(v) \cdot X = \int_{v \in V} \delta(v)$</td>
</tr>
<tr>
<td>AT3</td>
<td>$\int_{v \in V} \delta(v) = \delta(\sup(V))$</td>
</tr>
<tr>
<td>AT4</td>
<td>$t \leq \sup(V)$ $\int_{v \in V} a(v) + \delta(t) = \int_{v \in V} a(v)$</td>
</tr>
<tr>
<td>AT5</td>
<td>$\int_{v \in V} (a(v) \cdot X) = \int_{v \in V} (a(v) \cdot (v \gg X))$</td>
</tr>
<tr>
<td>ATB1a</td>
<td>$t \gg \int_{v \in V} a(v) = \int_{v \in V} (a(v) \cdot X) + \delta(t)$</td>
</tr>
<tr>
<td>ATB1b</td>
<td>$t \gg \int_{v \in V} a(v) \cdot X = \int_{v \in V} (a(v) \cdot X) + \delta(t)$</td>
</tr>
<tr>
<td>ATB3</td>
<td>$t \gg (X + Y) = (t \gg X) + (t \gg Y)$</td>
</tr>
<tr>
<td>ATB4</td>
<td>$t \gg (X \cdot Y) = (t \gg X) \cdot Y$</td>
</tr>
</tbody>
</table>

Table 11: BPAr6I = BPA+INT1,5+AT1,5+ATB1+ATB3,4
5.3 The Theory of BPAρI

INT1 This axiom expresses that an integration can be considered as an alternative composition over a continuum of alternatives. Note that terms are only identified in case V₁ ∪ V₂ is a well defined interval.

INT2 Choosing from the empty set is not defined and corresponds with a deadlock.

INT3 We may widen the scope if the leading time variable v of the first part of a sequential composition does not occur in the tail of this sequential composition.

INT₄ This axiom is our equivalent of the [BB89] axioms

\[(∀t ∈ V : P(t) = Q(t)) \implies \int_{v ∈ V} P = \int_{v ∈ V} Q\]

ATI₁ Though this axiom looks exactly like ATAI it is not called ATAI because a(0) is an abbreviation of \(\int_{v ∈ [0,0]} a(v)\)

ATB₂ This is the integral equivalent of both ATB₁ and ATB₂.

All other axioms are the straightforward equivalents of the BPAρₓ axioms.

Removing the Scope Brackets

According to the scope convention we can write \(\int_{v ∈ <1,2> \cdot \int_{v ∈ <3,4>}{(a(w))}}\) as \(\int_{v ∈ <1,2>}{a(v) \cdot \int_{v ∈ <3,4>}{b(w)}}\) and no scope brackets are needed. For other terms however we have to do some work before all scope brackets can be removed. Consider the following term, where the time variable v of the last integral is in the scope of the first one.

\[\int_{v ∈ <0,1>} (c(v) \cdot \int_{v ∈ <1,2>}{a(v) \cdot \int_{v ∈ <3,4>}{b(w) \cdot \int_{v ∈ <5,6>}{(d(u))}}})\]

Apply an α-conversion

\[\int_{v' ∈ <0,1>} (c(v') \cdot \int_{v ∈ <1,2>}{a(v) \cdot \int_{v ∈ <3,4>}{b(w) \cdot \int_{v ∈ <5,6>}{d(u)}}})\]

and a scope widening according to INT₃ₐ,ₖ

\[\int_{v' ∈ <0,1>} (c(v') \cdot \int_{v ∈ <1,2>}{(a(v) \cdot \int_{v ∈ <3,4>}{b(w) \cdot \int_{v ∈ <5,6>}{d(u)}}}))\]

before removing the brackets according to the scope convention

\[\int_{v' ∈ <0,1>} c(v') \cdot \int_{v ∈ <1,2>}{a(v) \cdot \int_{v ∈ <3,4>}{b(w) \cdot \int_{v ∈ <5,6>}{d(u)}}}\]

In order to make this more formal we need a definition and a Lemma.

**Definition 5.3.1** The set of widest scope terms \(W\) is defined as follows

\[W = \{p | p \text{ does not contain a subterm of the form } p₀ \cdot p₁\}\]

**Lemma 5.3.2** Every term \(p ∈ T\) has a widest scope term \(p_w ∈ W\) such that

\[\text{INT₃} ⊢ p = p_w\]

**Proof.** By induction on the size of \(p\), using α-conversion and scope widening.

**Remark 5.3.3** We can write all widest scope term from \(W\) without scope brackets.

In the following we will not use the scope brackets any more, since we may restrict ourselves to widest scope terms.
5.4 Basic Terms

In Definition 2.2.1 the notion of a basic term for BPAp6 (without integration) is introduced. In this section we introduce a notion of basic terms for BPAp6I. Before giving the formal definition we start with some informal remarks.

Remember that the idea of a basic term is, that it does not contain “redundant” information, for example:

\[
\int_{v \in <0,10>} a(v) \cdot \{\int_{w \in <0,10>} b(w) + \int_{z \in <0,10>} \delta(z)\} = \int_{v \in <0,10>} a(v) \cdot \int_{w \in v,10} \delta(w)
\]

\[
a(10) \cdot \{\int_{v \in <0,20>} b(v) + \delta(5) + \int_{w \in <0,31>} \delta(w)\} = a(10) \cdot \{\int_{v \in <10,20>} \delta(v) + \delta(30)\}
\]

The terms on the left-hand side are not basic terms, since they contain “redundant” information; intervals can be decreased and summands can be removed. The terms on the right-hand side are basic terms. Another characterisation of basic terms is, that every subinterval corresponds with a set of transitions in the associated transition system. Or, in other words, for basic terms there is a strong correspondence between the intervals and bounds on the one hand, and the set of timestamps of the transition system on the other hand. Indeed, in our completeness proof at the end of this section we will restrict ourselves to basic terms, using a Lemma of this section, which states that every term does have a basic term.

Definition 5.4.1 Let \( p, q \in W, V \in \text{Int}_{time-closed}, t \in \mathbb{R}^{\geq 0}, r, s \in \mathbb{R}^{\geq 0} \) and \( v \in TVar \).

\( B, B^*(t) \) and \( B^+(t) \) are the smallest sets satisfying

\[
\begin{align*}
& t \leq V \\
& t < s \\
& t < V \wedge (\forall r \in V \setminus \{v\}, p[r/v] \in B(r) \wedge p \simeq \delta) \quad \int_{v \in V} a(v) \cdot p \in B^+(t) \\
& p \in B^+(t) \wedge q \in B^+(t) \wedge (r = s \vee (r < s = \omega \wedge r > U(q))) \quad p + q, q + p \in B^+(t) \\
& p \in B^+(t) \wedge \delta, \delta + p \in B^+(t)
\end{align*}
\]

\[
B(t) = U_{s \in \mathbb{R}^{\geq 0}} B^+(t) \\
B = B(0) \cup \{\}
\]

Note that \( B \subseteq T^{time-closed} \). Until now we have dealt only with time-closed terms. But if we want to prove by induction that every term has a basic term, we have to consider terms with free time variables as well. Therefore, we first introduce the notion of a conditional term. A conditional term is some construct which determines for every substitution of real values for the free time variables a time-closed term. For example, if we consider the term \( a(5) \cdot b(v) \), which has a free time variable \( v \), we will associate to it the following conditional term:

\[
\begin{align*}
& \text{if the context assigns a value } t \leq 5 \text{ to } v \text{ then deliver } a(5) \cdot \delta \\
& \text{if the context assigns a value } t > 5 \text{ to } v \text{ then deliver } a(5) \cdot b(v)
\end{align*}
\]

In the sequel a conditional term is denoted as follows (the notation \( \rightarrow \) is taken from [BB90]):

\[
\begin{align*}
& \{v \leq 5 \rightarrow a(5) \cdot \delta\} \\
& p \simeq + \\
& \{v > 5 \rightarrow a(5) \cdot b(v)\}
\end{align*}
\]

For example, the substitution \( c \equiv [1/v] \) validates \( v \leq 5 \), so \( c(p) = a(5) \cdot \delta \) which is a basic term. Now consider a substitution \( \sigma' \equiv [6/v] \). Then \( \sigma' \) validates \( v > 5 \), so \( \sigma'(p) = a(5) \cdot b(6) \) which is again a basic term. For every other substitution \( \sigma'' \) which assigns a real number to \( v \) we have \( \sigma''(p) \in B \). In the following we introduce a generalisation of a basic term, called a conditional basic term. The idea
5.4 Basic Terms

is that a conditional term is called basic if any substitution of reals for the free time variables applied on the conditional term gives a basic term.

In the sequel we introduce a machinery by which we can state that the term \( w \models \int_{v \in <v_0, v_1>} a(v) \) with \( w, v_0, v_1 \in TVar \) has the following conditional basic term \( z \) associated with it:

\[
\begin{align*}
\{ w_0 \geq v_1 \land w \geq v_1 \} & : \rightarrow \ \delta(w) \\
+ \\
\{ w_0 < w < v_1 \} & : \rightarrow \ \int_{v \in <v_0, v_1>} a(v) \\
+ \\
\{ w \leq v_0 < v_1 \} & : \rightarrow \ \int_{v \in <v_0, v_1>} a(v)
\end{align*}
\]

or, if we take \( V = <v_0, v_1> \)

\[
\begin{align*}
\{ w \geq \text{sup}(V) \} & : \rightarrow \ \delta(w) \\
+ \\
\{ w < \text{sup}(V) \} & : \rightarrow \ \int_{v \in V \cap <w, w>} a(v)
\end{align*}
\]

Logical expressions like \( w \leq v_0 < v_1 \) are called conditions. Every condition represents a set of substitutions. In the example above there are three conditions, which together form a partition. Consider a substitution \( \sigma \) with \( \sigma(v_0) = 1, \sigma(w) = 2 \) and \( \sigma(v_1) = 3 \). Then the second condition, \( v_0 < w < v_1 \), is validated by \( \sigma \) (later we will say that \( \sigma \in [v_0 < w < v_1] \)). The associated term to this condition on the left side of the \( \models \) is \( \int_{v \in <w, v_1>} a(v) \), e.g.

\[
\begin{align*}
\{ 1 > 2 \land 2 \leq 1 \} & : \rightarrow \ \delta(w) \\
+ \\
\sigma(z) & \simeq \sigma( \{ 1 < 2 < 3 \} : \rightarrow \ \int_{v \in <2,3>} a(v) ) , \ \overset{1 < 2 < 3 = t} \simeq \ \int_{v \in <2,3>} a(v) \in B(2) \\
+ \\
\{ 2 \leq 1 < 3 \} & : \rightarrow \ \int_{v \in <1,3>} a(v)
\end{align*}
\]

\[ \implies \sigma(z) \in B(\sigma(w)) \]

Conditional Terms

All these new notions are defined more formally in the following definitions:

- A substitution \( \sigma \in \Sigma \) is a function which assigns to each time variable a real number or the time variable itself. On \( \mathbb{R}^{2^\ast} \) each substitution \( \sigma \) behaves as the identity. If \( \sigma \in \Sigma \), then we define \( \sigma \in T \rightarrow T \) by induction on the size of \( p \). Moreover, we define \( \sigma : \text{Int} \rightarrow \text{Int} \).

\[
\begin{align*}
\Sigma^\ast & = (TVar \cup \mathbb{R}^{2^\ast}) \rightarrow \text{Int} (TVar \cup \mathbb{R}^{2^0}) \\
\Sigma & = \{ \sigma \in \Sigma^\ast \mid \forall t \in \mathbb{R}^{2^0} \ \sigma(t) = t \land \\
& \forall v \in TVar \ \sigma(v) > 0 \land (\sigma(v) \in TVar \implies \sigma(v) = v) \}
\end{align*}
\]

\[
\begin{align*}
\sigma_{v\nu} & = \lambda w. \text{if } v = w \text{ then } v \text{ else } \sigma(w) \\
\sigma(\{ v_0, v_1 \}) & = < \sigma(v_0), \sigma(v_1) > \\
\sigma(\int_{v \in V} a(v) \cdot p) & = \int_{v \in \sigma(v)} a(v) \cdot \sigma(v)(p)
\end{align*}
\]

- Some subsets of \( \Sigma \) may be represented by a condition. The set \( \text{Cond}^{at} \) of atomic conditions is as follows:

\[
\text{Cond}^{at} = \{ v_0 < v_1, v_0 = v_1 | v_0, v_1 \in \text{Bound} \} \cup \{ tt, ff \}
\]

An atomic condition is a linear equation or inequality over time variables and real numbers. Let \( \text{tvar}(\alpha) \) denote the set of time variable occurring in \( \alpha \). Hence, if \( \alpha \) is an equation from \( \text{Con}^{at} \) and \( v \in \text{tvar}(\alpha) \) then we can write \( \alpha \) as \( v = w \) for some \( w \in \text{Bound} \).
The set conditions consists of atomic conditions combined by logical connectives ($\alpha^a$ is taken from $\text{Cond}^a$):
\[
\alpha \in \text{Cond} \quad \alpha := \alpha^a \mid \alpha \land \alpha \mid \alpha \lor \alpha \mid \neg(\alpha)
\]

In $\text{Cond}$ we distinguish a subset $\text{Cond}^a$, the conditions in normal form. A condition is in normal form if it is a conjunction of atomic conditions. We denote the set of time variables of $\alpha \in \text{Cond}$ by $\text{tvar}(\alpha)$. We abbreviate a set of conditions $\{\alpha_1, ..., \alpha_n\}$ by $\overline{\alpha}$. Similarly, $\text{tvar}(\overline{\alpha})$ is an abbreviation for $\text{tvar}(\alpha_1) \cup ... \cup \text{tvar}(\alpha_n)$.

- Every $\alpha \in \text{Cond}$ stands for a subset of substitutions for the time variables of $\text{TVar}$, so:

\[
\begin{align*}
[t] &= \Sigma \\
[f] &= \emptyset \\
[v < w] &= \{\sigma \in \Sigma \mid \sigma(v) < \sigma(w)\} \\
[w = w] &= \{\sigma \in \Sigma \mid \sigma(v) = \sigma(w)\} \\
[\alpha \lor \beta] &= [\alpha] \cup [\beta] \\
[\alpha \land \beta] &= [\alpha] \cap [\beta] \\
[\neg(\alpha)] &= [\Sigma] - [\alpha]
\end{align*}
\]

Earlier we introduced a notion of equality and an ordering on $\text{Bound}$. For example, we have $v + 1 > v$ and $w = w$ for $w + 1 \neq v$. Hence, $[v + 1 > v] = [\Sigma]$ and $[w = w]$ if $\alpha \in \text{Cond}^a$ and $[\alpha] \neq [tt]$ then we have for all $\sigma \in [\alpha]$ that $\sigma(\text{tvar}(\alpha)) \subset \mathbb{R}^{>0}$.

- A set of conditions like $\{\alpha_1, ..., \alpha_n\}$ is called a partition if for every $\sigma \in \Sigma$ with $\sigma(\text{tvar}(\overline{\alpha})) \subset \mathbb{R}^{>0}$ we have $\exists! \sigma \in [\alpha]$.

- $\mathcal{C}$, with typical elements $p_c$, is the set of conditional terms.

Let $p \in \mathcal{W}$, $\alpha \in \text{Cond}$ and $v \in \text{Bound}$

\[
p_c \in \mathcal{C} \quad p_c := p \mid \alpha \rightarrow p_c \mid p_c + p_c \mid \int_{w \in \mathcal{W}} (a(w) \cdot p_c) \mid v \searrow p_c
\]

We may abbreviate a set $\{p_1, ..., p_n\}$ of terms from $\mathcal{C}$ by $\overline{p}$. If $\sigma \in \Sigma$ then we define $\sigma : \mathcal{C} \rightarrow \mathcal{C}$ as expected. We have

\[
\sigma(\alpha \rightarrow p) = \begin{cases} 
\sigma(p) & \text{if } \sigma \in [\alpha] \\
\neg & \text{otherwise}
\end{cases}
\]

On $\mathcal{C}$ the equivalence $\simeq$ means syntactic equivalence modulo $\alpha$-conversion, associativity and commutativity of the + and modulo equivalence on conditions. Hence, if $[\alpha] = [\beta]$ then $\alpha \rightarrow p \simeq \beta \rightarrow p$. Equivalences on $\mathcal{C}$ may be hard to read, e.g. $v = w \rightarrow p = w = w \rightarrow p$. Therefore, we write conditional terms between brackets. For example $\{v = w \rightarrow p\} = \{v = v \rightarrow p\}$. These brackets are not part of the syntax but are meta-brackets.

- Instead of $\{v < w \rightarrow p + v\} = \{w \rightarrow p\}$ we write $\{v \leq w \rightarrow p\}$. Furthermore, we denote $\{v_0 < v \leq v_1 \rightarrow p\}$ by $\{v \in \overline{v_0, v_1} \rightarrow p\}$. Hence, conditions may contain subconditions of the form $v \in V$ for arbitrary $v \in \text{Bound}, V \in \text{Int}$. Thus, if $v \in \text{TVar}$ and $V \in \text{Int}$ then the atomic condition $v \in V$ must be read as "the set $\gamma$ of substitutions such that for each $\sigma$ in $\gamma$ we have $\sigma(v) \in \sigma(V)$".

- In Table 12 we state identities over $\mathcal{C}$, resulting in an equational theory $\text{BPA}_6\mathcal{C}$ if combined with the axioms of $\text{BPA}_6$. A substitution can be seen as an abstraction of a context. We define the set on contexts, $\text{Context}$, where the empty context is denoted by $[]$ and $p$ is taken from $\mathcal{T}$.

\[
\mathcal{C}[] \in \text{Context} \quad \mathcal{C}[] := [] \mid \mathcal{C}[] + p \mid p + \mathcal{C}[] \mid \int_{v \in \mathcal{V}} (a(v) \cdot \mathcal{C}[]) \mid v \searrow \mathcal{C}[]
\]
5.4 Basic Terms

$Context^{time-closed}$ is the set containing all $time-closed$ contexts. Every $C[]$ has exactly one occurrence of $[,]$, which may be in the scope of some integrals binding $time$ variables, which are called the bound variables of $C[]$. Thus, $B^V([]) = \emptyset$ and $BV(\int_{v \in V} (a(v) \cdot C[])) = \{v\} \cup BV(C[])$. For every $p \in T$, and $C[] \in Context$ we can construct a term $C[p] \in T$. Note that $[p] \equiv p$.

<table>
<thead>
<tr>
<th>COND1</th>
<th>${tt \rightarrow p} = p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>COND2</td>
<td>${ff \rightarrow p} = \delta$</td>
</tr>
<tr>
<td>COND3</td>
<td>${\alpha \rightarrow (\beta \rightarrow p)} = {\alpha \land \beta \rightarrow p}$</td>
</tr>
<tr>
<td>COND4</td>
<td>${\alpha \rightarrow p + q} = {\alpha \rightarrow p} + {\alpha \rightarrow q}$</td>
</tr>
<tr>
<td>COND5</td>
<td>${\alpha \lor \beta \rightarrow p} = {\alpha \rightarrow p} + {\beta \rightarrow p}$</td>
</tr>
<tr>
<td>COND6</td>
<td>$r \Rightarrow {\alpha \rightarrow p} = {\alpha \rightarrow r \Rightarrow p}$</td>
</tr>
<tr>
<td>COND7</td>
<td>$[\alpha] = [\beta] \implies {\alpha \rightarrow p} = {\beta \rightarrow p}$</td>
</tr>
</tbody>
</table>

| COND8a | $\{c \geq \text{sup}(V) \rightarrow cl \gg \int_{v \in V} a(v)\} = \{c \geq \text{sup}(V) \rightarrow \ell(cl)\}$ |
| COND8b | $\{c \geq \text{sup}(V) \rightarrow cl \gg \int_{v \in V} a(v) \cdot p\} = \{c \geq \text{sup}(V) \rightarrow \delta(cl)\}$ |

$V_\ell = V \cap <c, \omega>$

| COND9a | $\{c \leq \text{sup}(V) \rightarrow cl \gg \int_{v \in V} a(v)\} = \{c \leq \text{sup}(V) \rightarrow \int_{c \in V'} a(v)\}$ |
| COND9b | $\{c \leq \text{sup}(V) \rightarrow cl \gg \int_{v \in V} a(v) \cdot p\} = \{c \leq \text{sup}(V) \rightarrow \int_{c \in V'} a(v) \cdot p\}$ |

| COND10a | $\{V = \emptyset \rightarrow \int_{v \in V} a(v)\} = \{V = \emptyset \rightarrow \emptyset\}$ |
| COND10b | $\{V = \emptyset \rightarrow \int_{v \in V} a(v) \cdot p\} = \{V = \emptyset \rightarrow \emptyset\}$ |

| COND11 | If $(\alpha_1 \land v \in W_1), \ldots, (\alpha_{\# I} \land v \in W_{\# I})$ is a partition and $\gamma \not\in tvar(\alpha)$ then $\int_{v \in V} a(v) \cdot (\Sigma_1(\alpha_1 \land v \in W_1 \rightarrow p_1)) = \Sigma_1(\alpha_1 \rightarrow \int_{v \in V \land W_1} a(v) \cdot p_1)$ |

Table 12: BPA\deltaIC = BPA\deltaI + COND 1-7 + COND 3-11

The justification of these conditional axioms is given by the following Lemma. Note that it is possible now to reason with terms, even if they contain free $time$ variables, by using these conditional axioms. Moreover, this Lemma implies that all the conditional axioms together have the same expressive power as the axiom INT5 (as long as we restrict ourselves to prefixed integration).

Lemma 5.4.2 $\forall p, q \in T$

BPA\deltaIC $\vdash p = q \iff \forall C[] \in Context^{time-closed}$ with $FV(p+q) \subseteq BV(C[])$ BPA\deltaI $\vdash C[p] = C[q]$

Proof. By induction on the depth of $[,]$ in $C[]$ using INT5.

In order to use COND11 we need the following Lemma which states that is possible to isolate all occurrences of a variable from $TVar$ in $\alpha$, by rewriting $\alpha$ into one or several conditions of the form
\[ \beta_i \land v \in V_i. \text{ For example, using the next Lemma we can derive that} \]
\[
\begin{array}{c}
p = \{ v < 3 \land v < w \land 1 < v :\rightarrow q \} \\
\{ 3 < v \land v = w \land v < z :\rightarrow q' \} \\
\{ v \leq 3 \land v \in 1, w :\rightarrow q \} \\
\{ 3 < w \land w < z \land v \in [w, w] :\rightarrow q' \}
\end{array}
\]
\]

Now, if we consider the term \( \int_{v \in \langle w, w \rangle} a(v) \cdot p \) we can "push" the conditions upwards by applying COND11.

**Lemma 5.4.3** \( \forall \alpha \in \text{Cond}^n \forall p \in C \forall v \in TVar \quad \exists \beta \subset \text{Cond}^n \text{ with } v \notin \text{tvar}(\beta) \exists V \subset \text{Int} \]
\[
\text{BPA}_R \vdash \{ \alpha :\rightarrow p \} = \sum_j \{ \beta_j \land v \in V_j :\rightarrow p \}
\]

**Proof.** We denote that part of \( \alpha \) in which \( \tau \) does not occur by \( \alpha \wedge v \). We abbreviate \( \alpha_1 \land \ldots \land \alpha_n \) by \( \land_{i \in \{1, \ldots, n\}} \alpha_i \), where \( \land_{i \in \emptyset} \alpha_i = \tau \).

- \( v \notin \text{tvar}(\alpha) \)

\[
\{ \alpha :\rightarrow p \} \simeq \{ \alpha \land v \in \mathbb{R}^\omega :\rightarrow p \}
\]

- \( v \) occurs in \( \alpha \) in a simple condition of the form \( v = c \) for some \( c \in \text{Bound} \).

\[
\alpha = \{ \alpha \wedge \land_{i \in I} v > a_i \land \land_{j \in J} v < b_j \land \land_{k \in K} v = w_k \} \text{ for some } I, J, K \neq \emptyset
\]
We take \( c = w_k \) for some \( w_k \) and substitute
\[
\alpha = \{ \alpha \wedge \land_{i \in I} c > a_i \land \land_{j \in J} c < b_j \land \land_{k \in K} c = w_k \land v = c \}
\]
thus for some \( \beta \) in which \( v \) does not occur we have
\[
\alpha = \{ \beta \land v = c \}
\]

Hence,
\[
\{ \alpha :\rightarrow p \} \simeq \{ \beta \land v \in [c, c] :\rightarrow p \}
\]

- \( v \) occurs in \( \alpha \) only in inequalities.

\[
\alpha = \{ \alpha \wedge \land_{i \in I} v > a_i \land \land_{j \in J} v < b_j \}
\]
for some \( I, J \) with \( I \cup J \neq \emptyset \). This condition can be rewritten as follows
\[
\alpha = \bigvee_{i,j \in I \times J} \{ \alpha \land v \land v \in [a_i, b_j] \}
\]
Thus these are \( \gamma_{i,j} \) in which \( v \) does not occur, such that
\[
\alpha = \bigvee_{i,j \in I \times J} \{ \gamma_{i,j} \land v > a_i \land v < b_j \}
\]
\[
\{ \alpha :\rightarrow p \} = \sum_{i,j \in I \times J} \{ \gamma_{i,j} \land v \in [a_i, b_j] :\rightarrow p \}
\]

\[ \square \]

**Corollary 5.4.4** \( \forall p \in C \forall v \in TVar \)
\[
p \simeq \sum_i \{ \alpha_i :\rightarrow p_i \};
\]
\[
\exists \beta \subset \text{Cond} \land v \notin \text{tvar}(\beta) \exists V \subset \text{Int} \exists \eta \subset C \quad p \simeq \sum_j \{ \beta_j \land v \in V_j :\rightarrow q_j \}
\]
We define a notion of basic conditional terms:

**Definition 5.4.5** \( v \in \text{Bound} \)

\[
\sum_i \{ \alpha_i \Rightarrow p_i \} \in B(v) \overset{\text{def}}{=} \begin{cases} \bar{\alpha} \text{ is a partition} \\ \bar{\forall} \sigma \in [\alpha_i] \Rightarrow \sigma(p_i) \in B(\sigma(v)) \cup \{ \delta \} \end{cases}
\]

For example, \( \int_{x<z,x+1>} a(v) \cdot \int_{y<z,10>} b(x) \in B(z) \) but \( \int_{y<z,10>} a(v) \notin B(z) \). Note that \( \int_{x<z,10>} a(v) \) can be rewritten to the (conditional) basic term \( \{ z < 10 : \int_{y<z,10>} a(v) \} + \{ z \geq 10 : \delta(\text{cl}) \} \).

The next Lemma states that every term in \( T \) (with possibly free time variables) can be rewritten to a basic (conditional) term.

**Lemma 5.4.6** \( \forall p \in C \forall cl \in \text{Bound} \) : \( \exists b_{p,cl} \in B(cl) \) with

\[
FV(b_{p,cl}) \subseteq FV(cl \gg p) \quad \land \quad BPA_\rho C + cl \Rightarrow p = b_{p,cl}
\]

**Proof.** By induction on the size of \( p \). Again, \( V_{cl} \) is an abbreviation of \( V \cap < cl, > \). Note that it would not be possible to give this proof if we would integrate over arbitrary subsets of \( \mathbb{R}^{>0} \), as is the case in general integration.

- \( \int_{x \in V} a(v) \)

\[
cl \gg \int_{x \in V} a(v) = \{ cl \geq \sup(V) : \Rightarrow \delta(cl) \} + \{ cl < \sup(V) : \Rightarrow \int_{x \in V} a(v) \}
\]

- \( p + q \)

\[
cl \gg (p + q) = (cl \gg p) + (cl \gg q) = \sum_i \{ \alpha_i : p_i \} + \sum_j \{ \beta_j : q_j \} \quad \text{by induction}
\]

\[
= \sum_{(i,j)} \{ \alpha_i \land \beta_j : p_i + q_j \} \quad \text{since } \bar{\alpha} \text{ and } \bar{\beta} \text{ are partitions}
\]

\[
= \sum_{(i,j)} \begin{cases} \alpha_i \land \beta_j \land U(p_i + q_j) > U(p_i + q_j) : \Rightarrow p_i + q_j \\ \alpha_i \land \beta_j \land U(p_i + q_j) \leq U(p_i + q_j) : \Rightarrow p_i + q_j \end{cases} + \text{where } \bar{p} \text{ is the term } p \text{ without } \delta\text{-summands}
\]

\[
= \sum_{(i,j)} \begin{cases} \alpha_i \land \beta_j \land U(p_i + q_j) > U(p_i + q_j) : \Rightarrow p_i + q_j + \delta(U(p_i + q_j)) \\ \alpha_i \land \beta_j \land U(p_i + q_j) \leq U(p_i + q_j) : \Rightarrow p_i + q_j \end{cases}
\]
\[ \int_{v \in V} a(v) \cdot p \]

\[ cl \gg \int_{v \in V} a(v) \cdot p \]

\[ = \{ cl \geq \text{sup}(V) : \rightarrow \delta(cl) \} + \{ cl < \text{sup}(V) : \rightarrow \int_{v \in V} (a(v) \cdot \{v \gg p\}) \} \]

\[ = \{ cl \geq \text{sup}(V) : \rightarrow \delta(cl) \} + \{ cl < \text{sup}(V) : \rightarrow \int_{v \in V} (a(v) \cdot \sum_{i} \{ \alpha_i \land v \in W_i : \rightarrow p_i \}) \} \]

\[ \{ cl \geq \text{sup}(V) \} \quad \rightarrow \quad \delta(cl) \]

\[ + \quad \{ \sum_{i} \{ cl < \text{sup}(V) \land \alpha_i \land V_{cl} \land W_i = \emptyset \} \quad : \rightarrow \quad \delta(cl) \} \]

\[ + \quad \{ \sum_{i} \{ cl < \text{sup}(V) \land \alpha_i \land V_{cl} \land W_i \neq \emptyset \} \quad : \rightarrow \quad \int_{v \in V_{cl} \land W_i} a(v) \cdot p_i \} \} \}

Some of these \( p_i \)'s may be of the form \( \delta(v) \) and must be rewritten to \( \delta \). Since we have to end up with a basic term we have to construct a partition. This can be done by applying sufficiently many times the rules

\[ \{ \alpha \rightarrow p \} \quad \rightarrow \quad \{ \alpha \land \beta : \rightarrow p + q \} \]

\[ + \quad \{ \alpha \land \neg(\beta) : \rightarrow p \} \]

\[ \{ \beta \rightarrow q \} \quad \rightarrow \quad \{ \neg(\alpha) \land \beta : \rightarrow q \} \]

and

\[ \{ \alpha \rightarrow p + \delta(t) \} \rightarrow \{ \alpha \land t > U(p) \rightarrow p + \delta(t) \} + \{ \alpha \land t \leq U(p) \rightarrow p \} \]

\[ \bullet \tau \gg p \]

\[ cl \gg (\tau \gg p) \]

\[ = \quad cl \gg \sum_{i} \{ \alpha_i : \rightarrow p_i \} \quad \text{by induction} \]

\[ = \quad \sum_{i} \{ \alpha_i : \rightarrow cl \gg p_i \} \]

\[ = \quad \sum_{i} \{ \alpha_i : \rightarrow \sum_{j} (\beta_j : \rightarrow p_{i,j}) \} \]

\[ = \quad \sum_{i,j} \{ (\alpha_i \land \beta_j) : \rightarrow p_{i,j} \} \]

\[ \square \]

**Lemma 5.4.7** \( \forall p \in T \, \exists p_b \in C \) with \( \text{FV}(p_b) \subseteq \text{FV}(p) \land p_b \in B \) such that

\[ \text{BPA}_\phi \text{IC} \vdash p = p_b \]

**Proof.** Take \( p_b = b_{p,0} \)

\[ \square \]

**Theorem 5.4.8** \( \forall p \in T \) \text{time-closed} \( \exists p_b \in B \) such that

\[ \text{BPA}_\phi \text{IC} \vdash p = p_b \]
Proof. According to the previous Lemma there is a \( p_b \in B(0) \) with \( FV(p_b) \subseteq FV(p) = \emptyset \) and \( BPA_{\rho I} \vdash p = (0 \gg p) = p_b \). By Lemma 5.4.2 and \( FV[p + p_b] = \emptyset \) we know that for every context \( C[] \) we have \( BPA_{\rho I} \vdash C[p] = C[p_b] \). So we can take the empty context \( [] \) and the Theorem follows by \( \dashv \equiv \) for all \( \equiv \) in \( T \).

5.5 An Abstract Operational Semantics for BPA\( \rho I \)

In order to obtain action rules for BPA\( \rho I \), we have to adapt the action rules for the atomic cases for BPA\( \rho \). The other action rules can be taken from Table 5.

We expect the following transitions:

\[
\begin{align*}
&\int_{v \in <0,10>} a(v) \quad a(\delta) \quad \checkmark \\
&\int_{v \in <0,10>} a(v) \cdot \int_{w \in <v,v+1>} b(w) \quad a(\delta) \quad \int_{w \in <5,6>} b(w)
\end{align*}
\]

since \( \in <0,10> \). The Transition System Specification for BPA\( \rho I \) is given in Table 13. Bisimulation equivalence is denoted by \( \leftrightarrow \) and its definition can be found in Definition 3.3.1.

<table>
<thead>
<tr>
<th>Import rules w.r.t. ( \cdot + ) anc abs. shift from Table 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a \in A )</td>
</tr>
</tbody>
</table>
| \( \text{atom: } r \in V \setminus \{\#\} \) \( \int_{v \in V} a(v) \quad a(r) \quad \checkmark \\
\begin{align*}
&\int_{v \in V} (a(v) \cdot p) \quad a(r) \\
&\quad \exists \in p[\{v\}]
\end{align*}
| \( U(p) > L(p) \) |
| \( p \quad \delta(U(p)) \quad \checkmark 
| | |

Table 13: Transition System Specification for BPA\( \rho I \)

Theorem 5.5.1 The axioms of BPA\( \rho I \) are sound with respect to bisimulation equivalence.

Proof. Straightforward

Theorem 5.5.2 The operational semantics presented in this section for BPA\( \rho I \) is equivalent with the original semantics for BPA\( \rho I \):

\( \forall p, q \in T_{\text{time-closed}} \quad p \leftrightarrow q \iff <p,0> \iff <q,0> \)

Proof. Straightforward

The correspondence between the syntax of a basic term and its transition system can be stated as the Correspondence Lemma 3.3.4. Moreover, we have a Lemma similar to 3.3.5 stating that basic terms have ascending timestamps.
5.6 An Abstract Operational Semantics for BPAτδI in ntyft format

Now we have to introduce action rules in ntyft format which are equivalent with the action rules in Table 13. The rules for the atomic cases are straightforward translations of the *atom* rules of the Table 13. We can import the action rules in ntyft format w.r.t. · and (absolute) time shift from Table 4, where the action rules for BPAτδ are given. However, we cannot take the action rule for the + of Table 4, since we expect:

\[
\int_{v \in \llbracket 0,1 \rrbracket} a(v) + \delta(1) \xrightarrow{\delta(1)}
\]

because every *idle* step contributed by \(\delta(1)\) is contributed by \(\int_{v \in \llbracket 0,1 \rrbracket} a(v)\) as well.

In Table 14 a Transition System Specification in ntyft format is given. The transition relation is denoted by \(\mapsto\). Again we denote bisimulation equivalence for \(\mapsto\) by \(\leftrightarrow\). Note that

\[
\frac{U(p + q) > L(p + q)}{p + q \xrightarrow{\delta(U(p + q))} \delta} \quad \text{corresponds to the rule} \quad \frac{\{ \begin{array}{l} U(p) > L(p) \\ U(p) > L(q) \\ U(p) \geq U(q) \end{array} \} \quad \delta}{p + q \xrightarrow{\delta(U(p))} \delta, \quad q + p \xrightarrow{\delta(U(p))}}
\]

And this last rule corresponds in its turn to the \(\delta\) rule for the + of Table 14.

<table>
<thead>
<tr>
<th>Import rules w.r.t. · +, and abs. shift from Table 6, except the (\delta)-rule for the +</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a \in A)</td>
</tr>
<tr>
<td>(r \in V \quad \int_{v \in V} a(v) \xrightarrow{a(r)} \delta)</td>
</tr>
<tr>
<td>(r \in V \quad \int_{v \in V} (a(v) \cdot p) \xrightarrow{a(r)} r \gg p[r/v])</td>
</tr>
<tr>
<td>(sup(V) \neq 0)</td>
</tr>
<tr>
<td>(atom : \quad \int_{v \in V} \delta(v) \xrightarrow{\delta(sup(V))} \delta)</td>
</tr>
<tr>
<td>({ \begin{array}{l} p \xrightarrow{\delta(u)} p' \ r &lt; u \quad \forall r' \quad \forall b \in A \quad q \xrightarrow{b(r')} \end{array} ) \quad \delta(e)\</td>
</tr>
<tr>
<td>(+ : \quad \forall t \geq u \quad q \xrightarrow{\delta(t)} \quad \xrightarrow{p + \xi \delta(u)} {p'}_{\xi}, \quad q + p \xrightarrow{\delta(u)} p')</td>
</tr>
</tbody>
</table>

Table 14: Transition System Specification for BPAτδI in ntyft format

**Theorem 5.6.1** Bisimulation is a congruence
5.7 Completeness of BPAρI

Proof. See the proof of Theorem 3.4.2. The above rules are again in ntyfl format and the stratification of the proof of 3.4.2 remains valid.

Lemma 5.6.2 ∀p ∈ Ttime-closed ∀u ∈ R^2^0

\[ u = U(p) > L(p) \iff p \xrightarrow{\varepsilon_u} \delta \]

Proof. See Appendix C.

5.7 Completeness of BPAρI

Theorem 5.7.1 Completeness

∀p,q ∈ T

\[ p \leftrightarrow q \implies \text{EPAρI} \vdash p = q \]

We may restrict ourselves to basic terms, since every term is derivably equal to a basic term and since we have already proved soundness of BPAρI.

Proof. We prove that p ⊆ q by induction on the size of p. By symmetry, q ⊆ p follows and we are done. Assume

\[ p = \sum_i \int_{v_i \in V_i} a_i(v_i) \cdot p_i + \sum_j \int_{w_j \in W_j} b_j(w_j) \]

∀i ∀t ∈ V_i

\[ \begin{align*}
  & p \xrightarrow{a_i(t)} t \Rightarrow p_i[t/v_i] \\
  & q \xrightarrow{a_i(t)} t \Rightarrow q'
\end{align*} \]

By bisimulation q also has a transition

where

\[ \begin{align*}
  & t \Rightarrow p_i[t/v_i] \iff t \Rightarrow q' \\
  & p_i[t/v_i] \Rightarrow t \Rightarrow p_i[t/v_i]
\end{align*} \]

Since both p and q are basic terms

and

\[ t \Rightarrow q' \iff q' \Rightarrow q \\

p_i[t/v_i] = q' \]

Together with

\[ a_i(t) \cdot q' \subseteq q \]

we may conclude

\[ \text{(INT4)} \quad \int_{w \in V_i} a_i(w) \cdot p_i \subseteq q \]

∀j ∀t ∈ W_j

\[ \begin{align*}
  & p \xrightarrow{b_j(t)} \checkmark \\
  & q \xrightarrow{b_j(t)} \checkmark
\end{align*} \]

By bisimulation also q has a transition

and thus

\[ \text{(INT4)} \quad \int_{w \in W_j} b_j(w) \subseteq q \]

adding together: p ⊆ q

5.8 The Theory of ACPρI

In this section we incorporate the operators ||, | and ⊙ into the theory BPAρI. First we have to adapt the definition of our set of terms T by adding the rules

\[ p, q ∈ T, ⊂ ∈ \{||, |, ⊙\} \implies p ⊙ q ∈ T \]

\[ s ∈ R^2^0 \implies p \gg s ∈ T \]
to the definition of $T$ in Section 5.2. Next, the definitions of $FV()$ and substitution must be adapted. Moreover, we have to extend the definition of the syntactic operators $initact()$, $inittim()$ (see 4.1.2). We give only the new rules for the atomic cases, the other rules from 4.1.2 remain valid.

Let $V = \mathcal{I}^\text{time-closed}$

\[
\begin{align*}
\text{initact}(\int_{v \in V} \delta(v)) &= \emptyset \\
\text{initact}(\int_{v \in V} a(v)) &= \{a(t) | t \in V \setminus \{0\}\}
\end{align*}
\]

We introduce a theory $\ACP\rho$, which consists of adapted axioms of $\ACP\rho$ as given in Table 7.

\[
\begin{array}{l}
a, b \in A \quad V \in \mathcal{I}^\text{time-closed} \\
\text{ATBI5a} \quad r \gg \inf(V) \quad \int_{v \in V} a(v) \gg r = \int_{v \in V \cap <0, r>} a(v) \\
\text{ATBI5b} \quad r \gg \inf(V) \quad \int_{v \in V} (a(v) \cdot p) \gg r = \int_{v \in V \cap <0, r>} (a(v) \cdot p) \\
\text{ATBI6a} \quad r \leq \inf(V) \quad \int_{v \in V} a(v) \gg r = \delta(r) \\
\text{ATBI6b} \quad r \leq \inf(V) \quad \int_{v \in V} (a(v) \cdot p) \gg r = \delta(r) \\
\text{ATB7} \quad (X + Y) \gg t = (X \gg t) + (Y \gg t) \\
\text{ATB8} \quad (X \cdot Y) \gg t = (X \gg t) \cdot Y \\
\text{ATC1} \quad V_0 \cap V_1 \neq 0 \quad \int_{v \in V_0} a(v) \gg \int_{v \in V_1} b(v) = \int_{v \in V_0 \cap V_1} (a|b)(v) \\
\text{ATC2} \quad V_0 \cap V_1 = 0 \quad \int_{v \in V_0} a(v) \gg \int_{v \in V_1} b(v) = \delta(\min(\sup(V_0), \sup(V_1))) \\
\text{CM1} \quad X\|Y = X\|Y + Y\|X + X\|Y \\
\text{ATCM12} \quad v \notin FV(Y) \quad \int_{v \in V} (a(v))\|Y = \int_{v \in V} (a(v) \gg U(Y)) \cdot Y \\
\text{ATCM13} \quad v \notin FV(Y) \quad \int_{v \in V} (a(v) \cdot X)\|Y = \int_{v \in V} ((a(v) \gg U(Y)) \cdot (X\|Y)) \\
\text{CM4} \quad (X + Y)\|Z = X\|Y + Y\|Z \\
\text{CM5} \quad V_0 \cap V_1 \neq 0 \quad \int_{v \in V_0} (a(v) \cdot X) \gg \int_{v \in V_1} b(v) = \int_{v \in V_0 \cap V_1} ((a|b)(v) \cdot X) \\
\text{CM5'} \quad V_0 \cap V_1 = 0 \quad \int_{v \in V_0} (a(v) \cdot X) \gg \int_{v \in V_1} b(v) = \delta(\min(\sup(V_0), \sup(V_1))) \\
\text{CM6} \quad V_0 \cap V_1 \neq 0 \quad \int_{v \in V_0} a(v) \gg \int_{v \in V_1} (b(v) \cdot Y) = \int_{v \in V_0 \cap V_1} ((a|b)(v) \cdot Y) \\
\text{CM6'} \quad V_0 \cap V_1 = 0 \quad \int_{v \in V_0} a(v) \gg \int_{v \in V_1} (b(v) \cdot Y) = \delta(\min(\sup(V_0), \sup(V_1))) \\
\text{CM7} \quad V_0 \cap V_1 \neq 0 \quad \int_{v \in V_0} (a(v) \cdot X) \gg \int_{v \in V_1} b(v) \gg Y = \int_{v \in V_0 \cap V_1} ((a|b)(v) \cdot (X\|Y)) \\
\text{CM7'} \quad V_0 \cap V_1 = 0 \quad \int_{v \in V_0} (a(v) \cdot X) \gg \int_{v \in V_1} b(v) \gg Y = \delta(\min(\sup(V_0), \min(V_1))) \\
\text{CM8} \quad (X + Y)\|Z = X\|Y + X\|Z \\
\text{CM9} \quad X\|Y + Z = X\|Y + X\|Z \\
\text{D1} \quad a \notin H \quad \delta_H(a) = a \\
\text{D2} \quad a \in H \quad \delta_H(a) = \delta \\
\text{ATD} \quad \delta_H(a(t)) = \delta_H(a)(t) \\
\text{D3} \quad \delta_H(X + Y) = \delta_H(X) + \delta_H(Y) \\
\text{D4} \quad \delta_H(X \cdot Y) = \delta_H(X) \cdot \delta_H(Y)
\end{array}
\]

Table 15: $\ACP\rho$

**Theorem 5.8.1 Elimination Theorem**

For each term $p \in T$ there is a basic term $p_b$ such that $\text{BP}\rho\delta I \vdash p = p_b$
5.9 An Abstract Operational Semantics for ACP$\rho$I

We can gather all the action rules from the previous Tables:

<table>
<thead>
<tr>
<th>Import atom rules for $a \neq \delta$ from Table 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import rules for $a \neq \delta$ w.r.t. $\cdot$, $+$ and (absolute) time shift from Table 4</td>
</tr>
<tr>
<td>Import rules for $a \neq \delta$ w.r.t. $</td>
</tr>
</tbody>
</table>

$$\frac{U(p) > L(p)}{p \xrightarrow{\delta(U(p))} \square}$$

Table 16: Transition System Specification for ACP$\rho$I

Theorem 5.9.1 The axioms of ACP$\rho$I are sound with respect of the bisimulation equivalence $\Rightarrow$

Proof. Left to the reader

Theorem 5.9.2 The operational semantics presented in this section for ACP$\rho$I is equivalent with the original semantics for ACP$\rho$I:

$$\forall p, q \in T_{time-closed} \quad p \Rightarrow q \iff <p,0> \Rightarrow <q,0>$$

Proof. Left to the reader

5.10 An Abstract Operational Semantics for ACP$\rho$I in ntyft format

In this section we give a Transition System Specification for ACP$\rho$I in ntyft format. We can take almost all the action rules from the previous Tables. Only for the $\delta$-rules for $|$ and $\parallel$ we have to give new ones. For example, if we take the following terms with $a\in \notin \delta$ and $a\in \delta$, we do not expect $\delta$-transition in the first case but we do expect one in the second case.

$$\begin{align*}
\int_{e\in <0,1>} a(v) | \int_{e\in <0,1>} \bar{a}(v) & \xrightarrow{t(1)} \\forall
\int_{e\in <0,1>} a(v) | \int_{e\in <0,1>} \bar{a}(v) & \xrightarrow{t(1)} \sqrt{v}
\end{align*}$$

Note that the $\delta$-rule of Table 10 does not generate a $\delta$-transition in any of the two cases.
The point is, that the following implication does not hold anymore:

\[ U(p) = u \implies \exists a \in A_\delta \uplus A_\gamma \exists p' \in T \quad p \xrightarrow{a(u)} p' \]

Instead, now we have

\[
U(p) = u \iff \left\{ \begin{array}{l}
\exists V \sup(V) = u \forall s \in V \quad \exists a \in A_\delta \uplus A_\gamma \exists p' \in T \quad p \xrightarrow{a(s)} p' \\
\forall b \in A_\delta \uplus A_\gamma \forall r > u \quad p \not\xrightarrow{\delta(r)}
\end{array} \right. \}
\]

The obtained \( \delta \)-rules for \( \| \) and \( \perp \) are given in Table 17. As usual the transition relation is denoted by \( \xrightarrow{\rightarrow} \) and the bisimulation equivalence is denoted by \( \xrightarrow{\sim} \). Finally we can “import” all the

\[
\begin{array}{c}
\{ \sup(V) = u \forall s \in V \quad \exists a \in A_\delta \uplus A_\gamma \exists p' \quad p \xrightarrow{a(s)} p' \\
\forall b \in A_\delta \uplus A_\gamma \forall r > u \quad p \not\xrightarrow{\delta(r)} \\
\sup(W) \geq u \forall z \in W \quad \exists a \in A_\delta \uplus A_\gamma \exists q' \quad q \xrightarrow{a(z)} q' \\
\exists c' \in A_\delta \uplus A_\gamma \quad (p \xrightarrow{c'(s)} q \xrightarrow{c'(s)} q' \forall c' \delta(c') = \delta) \}
\end{array}
\]

\[
\begin{array}{c}
\sup(W) \geq u \forall z \in W \quad \exists a \in A_\delta \uplus A_\gamma \exists q' \quad q \xrightarrow{a(s)} q' \\
\forall b \in A_\delta \uplus A_\gamma \forall r > u \quad q \not\xrightarrow{\delta(r)} \\
\exists c' \in A_\delta \uplus A_\gamma \quad (p \xrightarrow{c'(s)} q \xrightarrow{c'(s)} q' \forall c' \delta(c') = \delta) \}
\end{array}
\]

Table 17: Additional \( \delta \)-rules for some merge-operators

action rules to obtain an operational semantics for ACP\( \rho \). This is given in Table 18

Theorem 5.10.1 Bisimulation equivalence is a congruence

Proof. Analogous to 4.4.1

\[
\textbf{Lemma 5.10.2} \quad \forall p \in \mathcal{T}_{\text{time-closed}} \quad \forall u \in \mathbb{R}^+ \\
\quad u = U(p) > \mathcal{L}(p) \iff p \xrightarrow{\delta(u)} \delta
\]
Import atom rules for $a \neq \delta$ from Table 14
Import rules for $a \neq \delta$ w.r.t. $\cdot$, $+$ and (absolute) time shift from Table 6
Import rules w.r.t. $\|\|$, $\ll$ and bounded initialization from Table 9

Table 18: Transition System Specification for ACP$\rho$I in ntyb format

Proof. See Appendix

Theorem 5.10.3 The two Transition System Specifications and the associated definitions of bisimulation induce the same equivalence.
$\forall p, q \in \mathcal{T}_{\text{time-closed}}$

\[
p \equiv q \iff p \equiv \cdots q
\]

Proof. See the proofs of Theorem 3.4.5 and Theorem 4.4.3

6 Concluding Remarks

In this paper we obtained a soundness and completeness result for Real Time Process Algebra with integration. We had to introduce some machinery, such as basic terms and conditional terms, to come this far. However, it is our belief that these introduced notions have a value in itself. The complexity we encountered was due to the notion of integration, which can be seen as an alternative composition over a continuum associated with a variable binding and scoping mechanism. We restricted ourselves to prefixed integration, thus not allowing general integration. However, we don't know any real-life processes which are ruled out by this restriction.

We did not consider other work like [Gro90], [MT90], [NS90], [RR88] or [Wan90]. Neither of these papers contain a notion similar to integration and most of the works are based on discrete time. Therefore, these works are less general and less complicated. However, it is an interesting question whether there are common ideas within the field of timed process algebra. For example, most authors believe that idling is non-deterministic. In our formalism this is formulated by $t \gg X + t \gg Y = \tau \gg \{(t - r) \gg X + (t - r) \gg Y\}$ if $r < t$. However, other authors like Groote, and until recently Sifakis as well, believe that idling forces a choice. It would be clarifying to have charted all these similarities and differences.

There are more concepts introduced or touched upon in [BB89] than treated in this paper. Topics for future research are recursion, abstraction and the relation between absolute and relative time. Another interesting question is, whether it is possible to give a finite equational logic and an operational semantics containing only finitely branching transition systems for terms with integration but without unguarded recursion. This would be an important step towards practical applications.

6.1 Acknowledgements

The author wants to thank, first of all, Jos Baeten for his patience in reading all previous versions and for his support and criticism. Thanks go to the author's colleagues Willem Jan Fokkink, Jan
Friso Groote, Henri Korver and Alban Ponce for discussing ideas and for giving help in \LaTeX, to Jan Bergstra for his comments on the presentation of the basic and conditional terms and to Frits Vaandrager for giving a first idea how to define the basic terms in the way they are defined now. Finally, Martin Wirsing is thanked for suggesting the research of completeness in $ACP\rho$ with integration.

References


A Soundness of the Axioms of BPA_δ

A.1 Proof of Theorem 3.2.4

Theorem 3.2.4

The axioms of BPA_δ are sound w.r.t. to the ultimate delay bisimulation equivalence \( \equiv_U \).

Proof.

- **A1 - A5**
  Trivial, soundness follows directly from the action rules for \( \cdot \) and \( + \).  

- **ATA1** \( a(0) = \delta(0) \)
  Both \( a(0) \) and \( \delta \) do not have any transitions, moreover \( U(a(0)) = U(\delta) = 0 \).

- **ATA2** \( \delta(t) \cdot X = \delta(t) \)
  Both \( \delta(t) \cdot X \) and \( \delta(t) \) do not have any transitions, moreover \( U(\delta(t) \cdot X) = U(\delta(t)) = t \).

- **ATA3** \( \delta(t) + \delta(\tau) = \delta(\tau) \) whenever \( t \leq \tau \)
  Both \( \delta(t) + \delta(\tau) \) and \( \delta(\tau) \) do not have any transitions, moreover \( U(\delta(t) + \delta(\tau)) = U(\delta(\tau)) = \tau \) whenever \( t \leq \tau \).

- **ATA4** \( a(t) + \delta(t) = a(t) \)
  Both terms have only one transition \( \frac{a(t)}{\ \sqrt{\ }} \), moreover \( U(a(t) + \delta(t)) = \max(U(a(t)), U(\delta(t))) = t = U(a(t)) \).

- **ATA5** \( a(t) \cdot X = a(t) \cdot (t \gg X) \)
  Note that \( U(a(t) \cdot X) = U(a(t)) = t = U(a(t) \cdot (t \gg X)) \) and both terms can only perform a transition \( \frac{a(t)}{\ \sqrt{\ }} \) resulting in resp. \( t \gg X \) and \( t \gg (t \gg X) \). Now we have to prove that \( t \gg X \) is bisimilar with \( t \gg (t \gg X) \). If \( t \gg X \xrightarrow{a(t)} X' \) then it must be the case that \( \tau > t \) and this implies \( t \gg (t \gg X) \xrightarrow{a(t)} X' \). Moreover, if \( U(X) < t \) then \( U(t \gg X) = U(t \gg (t \gg X)) \) and otherwise \( U(t \gg X) = U(t) \gg (t \gg X) \). The other direction, for which we have to prove that every transition of \( t \gg X \) has a corresponding transition in \( t \gg X \) can be done similarly.

- **ATB1** \( t < \tau \Rightarrow t \gg a(r) = a(\tau) \)
  Both terms have only the transition \( \frac{a(\tau)}{\ \sqrt{\ }} \), moreover since \( t < \tau \) we have \( U(t \gg a(\tau)) = \tau = U(a(\tau)) \).
\section*{B Some Remarks on the Format of the Action Rules}

An equivalence relation can model an equational theory only when it is a congruence w.r.t. to all operators. The equivalence on transition systems we are dealing with is the bisimulation equivalence. Following [Gro89] we can obtain bisimulation equivalence as a congruence by using only action rules obeying some conditions. In this section we introduce this elegant result, by discussing a format of action rules such that bisimulation equivalence on the resulting transition systems is automatically a congruence.

In Table 5 we gave an action rule for $\delta$-transitions ($\delta$-rule):

$$\frac{U(p) \succ L(p)}{p \overset{\delta(U(p))}{\longrightarrow} \checkmark}$$

It will be shown in this section that this rule is not in the right format, since the premise $U(p) \succ L(p)$ depends too much on the syntax of the terms. However we can give a $\delta$-rule for each operator, for example for the $+$:

$$\frac{p \overset{\delta(r)}{\longrightarrow} p' \forall a \in A, \forall r' \geq \tau \ q \overset{a(r')}{\longrightarrow} \forall s \succ \tau \ q \overset{\delta(s)}{\longrightarrow} p + q \overset{\delta(r)}{\longrightarrow} p'}{p + q \overset{\delta(r)}{\longrightarrow} p'}$$

which might be read as

$$\frac{p \overset{\delta(r)}{\longrightarrow} p' \ L(q) \succ \tau \ U(q) \leq \tau}{p + q \overset{\delta(r)}{\longrightarrow} p'}$$

\end{itemize}
Formally, the premise of an action rule consists of a set of positive premises, denoting the transitions that must hold for applying this action rule, and a set of negative premises, denoting the absence of certain transitions. Thus actually, the given action rule is an abbreviation for

\[
\frac{\{ p \xrightarrow{\delta^{(r)}} p' \} \cup \{ q \xrightarrow{k(r')} \ | \ b \in A_\delta, \ r' \geq r \} \cup \{ q \xrightarrow{l_s} \ | \ s > r \}}{p + q \xrightarrow{\delta(r)} p'}
\]

where we make the set of positive premises and the set of negative premises explicit.

Note that although \( \delta(1) \) has a \( \delta(1) \)-transition, \( a(2) + \delta(1) \) does not have this \( \delta(1) \)-transition since it has a \( a(2) \)-transition.

If we take a closer look at the format of the action rule we recognise term variables like \( p, q \) and a constructor like \(+\). Moreover, we have to introduce the definition of \( \xrightarrow{\cdot} \), the set of "negative" transitions.

Some Definitions w.r.t. action rules an negative premises

- The relation \( \xrightarrow{\cdot} \subseteq \text{State} \times \text{Label} \) is induced in a negative way by the relation \( \xrightarrow{\cdot} \)

\[
(\forall q \in \text{State} : (p, \lambda, q) \not\xrightarrow{\cdot} ) \iff (p, \lambda) \in \xrightarrow{\cdot}
\]

and we write \( p \xleftarrow{\lambda} \) for \( (p, \lambda) \in \xrightarrow{\cdot} \). Hence, \( p \xleftarrow{\lambda} \) can be read as "\( p \) does not have any \( \lambda \)-labelled transitions".

- \( \text{TermVar} \) is a set of variables (over \( T \)), of which the elements are called term variables.

- \( F \) is a set of constructors with typical elements \( f \) each having a rank \( r(f) \). In our case \( F \) contains the constructors \(+, \cdot\) with \( r(+) = r(\cdot) = 2 \) and for each \( t \in \mathbb{R}^{\geq 0} \) there is a unary constructor \( t \gg \) with \( r(t \gg) = 1 \). Note that we have changed the treatment of the (absolute) time shift a little, but in this context we have to deal with single sorted constructors.

- \( T(\text{TermVar}, F) \) are the terms constructed by taking variables from \( \text{TermVar} \) and combining them by constructors from \( F \).

- An action rule of the following form:

\[
\frac{\{ t_k \xrightarrow{a_k(r_k)} y_k \ | \ k \in K \} \cup \{ t_l \xrightarrow{b_l(u_l)} \ | \ l \in L \}}{f(x_1, \ldots, x_{r(f)}) \xrightarrow{c(s)} t'}
\]

is in ntyft format if it obeys the following conditions:

1. \( t_k, t_l, t' \in T(\text{TermVar}, F) \)
2. \( y_k, x_i \in \text{TermVar} \) with \( y_k \) and \( x_i \) are pairwise disjoint
3. \( f \in F \)
4. \( a_k, b_l, c \in A_\delta \) where \( A_\delta \) is the alphabet as defined earlier. The variables \( r_k, u_l \) and \( s \) are called time variables and are taken from a set \( TVar \).

- An instantiation of an action rule is obtained by substituting terms for the term variables and substituting real numbers for time variables.
A stratification \( S \) is a function which assigns to each transition an element in some totally ordered well founded set, let us say \( D \):

\[
\nu : \text{State} \times \text{Label} \times \text{State} \rightarrow D
\]

An action rule of the above form can be stratified if for each possible substitution \( \gamma \)

\[
\nu[\gamma( t \xrightarrow{a(p)} y)] \leq \nu[\gamma( f(x_1, \ldots, x_{n(f)})) \xrightarrow{c(s)} !)]
\]

\[
\forall t \in \mathcal{L}, \forall t_{\gamma}, \nu[\gamma( t \xrightarrow{b(t_{\gamma})} t_{\gamma})] < \nu[\gamma( f(x_1, \ldots, x_{n(f)})) \xrightarrow{c(t)} !)]
\]

A Transition System Specification is well-founded if every backward chain in its Variable Dependency Graph is finite. The set of nodes of this Graph is the set of variables occurring in a positive premise of one of the action rules of the Transition System Specification. There is an arrow from \( x \) to \( y \) if there is a positive premise \( t_k \xrightarrow{a_k(x_k)} y_k \) such that \( x \) is a variable occurring in \( t_k \) and \( y \) is the variable occurring \( y_k \).

Constructing a transition relation from action rules

A transition relation \( \mathcal{R} \) is "constructed" from a set of action rules by first initialising \( \mathcal{R}^0 \) with all instantiations of the axioms in the set of rules. Next we will apply the inference rules in the set; if there are transitions in \( \mathcal{R}^n \) such that the positive premise of an inference rule can be instantiated and there are no instantiations of the negative premises in \( \mathcal{R}^n \) we obtain the instantiation of the conclusion as a new element of \( \mathcal{R}^{n+1} \), where \( \mathcal{R}^n \subseteq \mathcal{R}^{n+1} \). In this way we can construct the fixed point \( \mathcal{R} \).

However we have to be careful, once we used the fact that a negative premise was valid for \( \mathcal{R}^i \) we may not add a transition in \( \mathcal{R}^j \) with \( j > i \) neglecting this negative premise. A stratification and the well-foundedness guarantees that it will be the case.

Action rules in \textit{ntyft}-format result in a congruence

Now we have defined all the preliminaries concerning action rules with negative premises and their stratifications, we can give the following theorem which is used in the proofs of the Lemmas 3.4.2 and 4.4.1.

Theorem B.0.1 Bisimulation equivalence is a congruence for each well-founded Transition System Specification in which all rules are in \textit{ntyft} format and in which all the action rules can be stratified.

Proof. See [Gro89] \( \square \)

C The Transition System in \textit{ntyft} format has Consistent \( \delta \)-transitions

C.1 Proof of Lemma 3.4.3 and Lemma 4.4.2

Lemma 3.4.3

\[
\forall p \in \mathcal{T} \forall u \in \mathbb{R}^0 \quad v = U(p) > L(p) \iff p \xrightarrow{\delta(u)} \delta
\]

Proof. By induction on the size of \( p \). We give only some cases, the other cases are left to the reader.
C.1 Proof of Lemma 3.4.3 and Lemma 4.4.2

\[ \bullet p \in A^0_{\text{ine}} \]

\[ U(p) > L(p) \]
\[ \iff \]
\[ p \equiv \delta(r) \]
\[ \iff \]
\[ p \equiv \delta(r) \land \delta(r) \xrightarrow{\delta(s)} \sqrt{\Box} \]

\[ \bullet p \equiv p + q \]

\[ U(p + q) > L(p + q) \]
\[ \iff \]
\[ \{ U(p + q) > L(p + q) \land \{ U(p + q) > L(p + q) \land U(p) \geq U(q) \} \lor U(q) \geq U(p) \} \]
\[ \iff \]
\[ \{ U(p + q) > L(p) \land U(p + q) > L(q) \land U(p) \geq U(q) \} \lor \ldots \]
\[ \{ \max(U(p), U(q)) > L(p) \land \max(U(p), U(q)) > L(q) \land U(p) \geq U(q) \} \lor \ldots \]
\[ \{ U(p) > L(p) \land U(p) > L(q) \land U(p) \geq U(q) \} \lor \ldots \]
\[ \{ p \xrightarrow{\delta(s)} \sqrt{\Box} \land \forall a \in A \cup A_\lor : \forall r \geq u \ a(r) \xrightarrow{\delta(s)} \land \forall b \in A_k \cup A_\lor : \forall r > u \ b(r) \xrightarrow{\delta(r)} \} \lor \ldots \]
\[ \exists u \ u \geq 0 \]
\[ \{ \exists p' : p \xrightarrow{\delta(s)} p' \land \forall a \in A \cup A_\lor : \forall r > u \ a(r) \xrightarrow{\delta(r)} \land \forall r > u \ q \xrightarrow{\delta(r)} \} \lor \ldots \]

\[ \bullet p \equiv p \parallel q \]

\[ U(p \parallel q) > L(p \parallel q) \]
\[ \iff \]
\[ \{ U(p) > L(p \parallel q) \land U(p) \leq U(q) \} \lor \{ U(q) > L(p \parallel q) \land U(q) \leq U(p) \} \]
\[ \iff \]
\[ \{ U(p) > L(p \parallel q) \land U(p) \leq U(q) \} \lor \{ U(p) = L(p) \land U(p) > L(q) \} \lor \{ U(p) \leq U(q) \land U(p) > L(p) \} \lor \ldots \]
\[ \alpha_1 \lor \alpha_2 \lor \ldots \]
The transition system in NTYFT format has consistent δ-transitions.

We continue with α₁ and α₂ separately.

\[
\begin{align*}
\{ & U(p) > L(p \parallel q) & \wedge \\
& U(p) \leq U(q) & \wedge \\
& U(p) = L(p) & \\
\}
\end{align*}
\]

\[\iff\]

\[\exists z \in A \cup A_{\gamma} : a(U(p)) \in \text{initact}(p) \wedge U(p) \notin \text{inittim}(p \parallel q) \wedge U(p) \leq U(q) \}
\]

\[\iff\]

\[\exists z \in A \cup A_{\gamma} : a(U(p)) \in \text{initact}(p) \wedge U(p) \notin \text{inittim}(p \parallel q) \wedge U(p) \notin \text{inittim}(p \parallel q) \wedge U(p) \notin \text{inittim}(p \parallel q) \wedge U(p) \leq U(q) \}
\]

\[\iff\]

\[\exists z \in A \cup A_{\gamma} : a(U(p)) \in \text{initact}(p) \wedge U(p) \notin \text{inittim}(p \parallel q) \wedge U(p) \in \text{inittim}(p \parallel q) \implies \{ U(q) \leq U(p) \iff U(p) \notin \text{inittim}(p \parallel q) \}
\]

\[\iff\]

\[\exists z \in A \cup A_{\gamma} : a(U(p)) \in \text{initact}(p) \wedge U(p) \leq U(q) \implies U(p) \notin \text{inittim}(p \parallel q) \wedge U(p) \leq U(q) \}
\]

\[\iff\]

\[\exists z \in A \cup A_{\gamma} : a(U(p)) \in \text{initact}(p) \wedge U(p) = U(q) \wedge U(p) \notin \text{inittim}(p \parallel q) \}
\]

\[\iff\]

\[\exists z \in A \cup A_{\gamma} \exists u > 0
\]

\[\{ \exists p' \quad p \xrightarrow{u} p' \quad \forall b \in A_{\delta} \cup A_{\gamma} \forall r > u \quad p \xrightarrow{b(r)} \}
\]

\[\exists a' \in A_{\delta} \cup A_{\gamma} \exists q' \quad q \xrightarrow{a'(s)} q' \quad \forall b \in A_{\delta} \cup A_{\gamma} \forall r > u \quad q \xrightarrow{b(r)} \]

\[\forall c, c' \in A_{\delta} \cup A_{\gamma} \quad \{ p \xrightarrow{c(s)} q \quad q \xrightarrow{c'(s)} \quad \forall c' \neq c \}
\]

And α₂:

\[
\begin{align*}
\{ & U(p) > L(p \parallel q) & \wedge \\
& U(p) \leq U(q) & \wedge \\
& U(p) > L(p) & \\
\}
\end{align*}
\]

\[\iff\]

\[\{ \quad U(p) > L(p) \quad \implies \quad U(p) > L(p \parallel q) \}
\]

\[\iff\]

\[\{ \quad U(p) > L(p) \quad \implies \quad U(p) > L(p \parallel q) \}
\]

\[\iff\]

\[\{ \quad U(p) > L(p) \quad \implies \quad U(p) > L(p \parallel q) \}
\]

\[\iff\]

\[\{ \quad U(p) > L(p) \quad \implies \quad U(p) > L(p \parallel q) \}
\]

\[\iff\]

\[\{ \quad U(p) > L(p) \quad \implies \quad U(p) > L(p \parallel q) \}
\]

\[\iff\]

\[\{ \quad U(p) > L(p) \quad \implies \quad U(p) > L(p \parallel q) \}
\]

\[\iff\]

\[\{ \quad U(p) > L(p) \quad \implies \quad U(p) > L(p \parallel q) \}
\]
C.2 Proof of Lemma 5.10.2

Lemma 5.10.2
\[ \forall p \in T \forall u \in \mathbb{R}^{\geq 0} \]
\[ u = U(p) > L(p) \iff p \xrightarrow{\delta(U(p))} \sqrt{\cdot} \]

Note that \( T \) is now the set of terms concerning integrals as defined in section 5.8. We give only the +. The rules for \( \perp \) and \( \| \) must be changed as well, but their derivations are left to the reader.

Proof. By induction on the size of \( p \).

- \( p = p + q \)

\[ U(p + q) > L(p + q) \]

\[ \iff \]
\[ \{ U(p) > L(p) \land U(p) > L(q) \land U(p) \geq U(q) \} \lor \ldots \]

\[ \iff \exists u > 0 \]
\[ \{ p \xrightarrow{\delta(u)} \sqrt{\cdot} \land u > \text{sup(inittim}(q)) \land u \geq \text{sup(active}(q)) \} \lor \ldots \]

\[ \iff \exists u > 0 \]
\[ \{ \exists p' : p \xrightarrow{\delta(u)} p' \land \exists r < u \forall r' \geq r \forall b \in A \cup A \lor q \xrightarrow{b(r')} \land \forall b \in A \cup A \lor \forall t > u \land q \xrightarrow{\delta(t)} \} \lor \ldots \]

\[ \iff \exists u > 0 \]
\[ \{ \exists p' : p \xrightarrow{\delta(u)} p' \land \exists r < u \forall r' \geq r \forall b \in A \cup A \lor q \xrightarrow{\delta(r')} \land \forall t > u \land q \xrightarrow{\delta(t)} \} \lor \ldots \]