The United Nations University

UNU/IIST

International Institute for Software Technology

Timed Frame Models for Discrete Time Process Algebras

J.A. Bergstra, C.A. Middelburg, B. Warinschi

October 1997

UNU/IIST Report No. 100

UNU/IIST

UNU/IIST enables developing countries to attain self-reliance in software technology by: (i) their own development of high integrity computing systems, (ii) highest level post-graduate university teaching, (iii) international level research, and, through the above, (iv) use of as sophisticated software as reasonable.

UNU/IIST contributes through: (a) advanced, joint industry–university advanced development projects in which rigorous techniques supported by semantics-based tools are applied in case studies to software systems development, (b) own and joint university and academy institute research in which new techniques for *(1) application domain* and computing platform modelling, *(2) requirements capture*, and *(3) software design & programming* are being investigated, (c) advanced, post-graduate and post-doctoral level courses which typically teach Design Calculi oriented software development techniques, (d) events [panels, task forces, workshops and symposia], and (e) dissemination.

Application-wise, the advanced development projects presently focus on software to support large-scale infrastructure systems such as transport systems (railways, airlines, air traffic, etc.), manufacturing industries, public administration, telecommunications, etc., and are thus aligned with UN and International Aid System concerns. UNU/IIST is a leading software technology centre in the area of infrastructure software development.

UNU/IIST is also a leading research centre in the area of Duration Calculi, i.e. techniques applicable to *real-time, reactive, hybrid & safety critical systems.* The research projects parallel and support the advanced development projects.

At present, the technical focus of UNU/IIST in all of the above is on applying, teaching, researching, and disseminating Design Calculi oriented techniques and tools for trustworthy software development. UNU/IIST currently emphasises techniques that permit proper development steps and interfaces. UNU/IIST also endeavours to promulgate sound project and product management principles.

UNU/IIST's primary dissemination strategy is to act as a clearing house for reports from research and technology centres in industrial countries to industries and academic institutions in developing countries. At present more than 200 institutions worldwide contribute to UNU/IIST's report collection while UNU/IIST at the same time subscribes to more than 125 international scientific and technical journals. Information on reports received (and produced) and on journal articles is to be disseminated regularly to developing country centres — which are then free to order a reasonable number of report and article copies from UNU/IIST.

Dines Bjørner, Director — 02.7.1992 – 01.7.1997
Zhou Chaochen, Director — 01.8.1997 – 31.7.2001

UNU/IIST Reports are either $\mathcal{R}$esearch, $\mathcal{T}$echnical, $\mathcal{C}$ompendia or $\mathcal{A}$dministrative reports:

$\boxed{\mathcal{R}}$ Research Report • $\boxed{\mathcal{T}}$ Technical Report • $\boxed{\mathcal{C}}$ Compendium • $\boxed{\mathcal{A}}$ Administrative Report

The United Nations University

UNU/IIST

International Institute for Software Technology

P.O. Box 3058
Macau

Timed Frame Models for Discrete Time Process Algebras

J.A. Bergstra, C.A. Middelburg, B. Warinschi

Abstract

A model for discrete time process algebra with relative timing is given by defining an interpretation of the constants and operators on timed frames. It is shown that the model which is obtained is isomorphic with a graph model for the same algebra. *Keywords & Phrases:* discrete time, frame algebra, process algebra, relative timing, timed frames.

Jan Bergstra is a Professor of Programming and Software Engineering at the University of Amsterdam and a Professor of Applied Logic at Utrecht University, both in the Netherlands. His research interest is in mathematical aspects of software and system development, in particular in the design of algebras that can contribute to a better understanding of the relevant issues at a conceptual level. He is perhaps best known for his contributions to the field of process algebra. E-mail: janb@fwi.uva.nl

Kees Middelburg is a Senior Research Fellow at UNU/IIST. He is on a two year leave (1996–1997) from KPN Research and Utrecht University, the Netherlands, where he is a Senior Computer Scientist and a Professor of Applied Logic, respectively. His research interest is in formal techniques for the development of software for reactive and distributed systems, including related subjects such as semantics of specification languages and concurrency theory. E-mail: cam@iist.unu.edu

Bogdan Warinschi is a Fellow at UNU/IIST. He is on a nine month leave (September 1996–May 1997) from Bucharest University, Romania, where he is an undergraduate student in the Faculty of Mathematics, Computer Science Department. His general research interest is in algebraic methods in Computer Science. E-mail: bogw@skylab.math.unibuc.ro

Contents

1 Introduction **1**

2 Relative discrete time process algebra **1**
2.1 Basic process algebra 2
2.2 Recursion 4

3 Graph model for discrete time process algebra without recursion **6**

4 Timed frames **9**

5 Frame model for discrete time process algebra without recursion **15**

6 Recursion in graphs and frames **23**
6.1 Graph model for $\mathrm{BPA}_{\mathrm{drt}}^{=}\mathrm{Lin}$ 24
6.2 Timed frame model for $\mathrm{BPA}_{\mathrm{drt}}^{=}\mathrm{Lin}$ 28

7 Conclusions and future work **30**

1 Introduction

Process algebras are models of axiom systems which can be constructed with several techniques. An important technique is to view the domain of a process algebra as a set of equivalence classes of transition systems. Operations of process algebra are then represented as transformations of transition systems. In recent years process algebras which involve time have been proposed to fulfill the need for formalisms able to deal with quantitative time aspects of systems. The option to represent time by non-negative reals and to have time stamps on actions is taken into account in [2] for ACP, in [9] for CCS and in [1] for CSP. Another option is to divide time into slices, thus giving the possibility to use an implicit or explicit time stamping mechanism that provides each action with the index of the time slice in which it occurs. This has been developed for the case of ACP in [3].

Frame algebra is developed in [8] in order to obtain an algebraic framework for transition systems quite independent of process algebra. This will allow to obtain algebraic representations of transformation of transition systems corresponding to operations of process algebra. Simple frames are built from states and action-labelled transitions. Equipped with a root marker and optionally with a termination marker they make up transition systems. A further step is taken in [7] where special transitions are introduced to model passage of (discrete) time, thus providing a framework in which one can deal with time aspects, others than precedence relations. In order to keep the basic operations on frames as simple as possible, and at the same time to provide the level of abstraction needed for studying discrete time processes a special kind of bisimulation, called σ-bisimulation, has been introduced. The objective of this paper is to generalise [8] to a setting of discrete time processes. For this purpose timed frames are needed which were introduced in [7]. We obtain a process algebra based on timed frames and explicit algebraic descriptions of the transformations that correspond to the process operations.

The structure of the paper is as follows. First of all, we give a survey of relative discrete time process algebra (Section 2). Further a graph model for discrete time process algebra without recursion is presented (Section 3). The next section is dedicated to an survey of timed frames (Section 4). Section 5 introduces a timed frame model for basic discrete time process algebra. Finally we construct a graph model and a frame model for basic discrete time process algebra with recursion and we prove that they are isomorphic (Section 6).

2 Relative discrete time process algebra

Process algebra in the form of ACP describes the main features of concurrent programs but does not deal explicit with time. However the time order is covered; $p \cdot q$ expresses that the process p has to be performed before the process q. A more quantitative view of time is taken into account in discrete time process algebra where a division of time in slices is used. For example, the property that a process is delayed n time slices can be expressed in this setting.

In Section 2.1 the theory of $\mathrm{BPA}_{\mathrm{drt}}^{=}$ is presented. This is the kernel of discrete relative time process algebra. The superscript $=$ stands for the absence of the immediate deadlock constant used in [4] to provide conformity in the absolute and relative time cases. Recursion is added in Section 2.2.

2.1 Basic process algebra

In this subsection we describe basic discrete relative time process algebra without immediate deadlock (notation $\mathrm{BPA}_{\mathrm{drt}}^{=}$).

It is assumed that a fixed but arbitrary set A of actions has been given, such that δ is not in A. We denote $\mathsf{A} \cup \{\sigma\}$ by A_σ.

The signature of $\mathrm{BPA}_{\mathrm{drt}}^{=}$ is as follows:

Constants:

$\mathsf{cts}(a)$	$a \in \mathsf{A}$	(a in the current time slice)
$\mathsf{cts}(\delta)$	deadlock ($\delta \notin \mathsf{A}$)	(deadlock in the current time slice)

Unary operators

σ_{rel}	delay operator

Binary operators:

$\cdot$	sequential composition
$+$	alternative composition

Given the signature, terms of $\mathrm{BPA}_{\mathrm{drt}}^{=}$, usually referred to as process expressions, are constructed in the usual way. We write $\mathcal{P}$ for the set of all variable-free process expressions. We shall use meta-variables x, x', y and y' to stand for arbitrary process expressions. The axioms of $\mathrm{BPA}_{\mathrm{drt}}^{=}$ are given in Table 1.

$x + y = y + x$	A1	$\sigma_{\mathsf{rel}}(x) + \sigma_{\mathsf{rel}}(y) = \sigma_{\mathsf{rel}}(x + y)$	DRT1
$(x + y) + z = x + (y + z)$	A2	$\sigma_{\mathsf{rel}}(x) \cdot y = \sigma_{\mathsf{rel}}(x \cdot y)$	DRT2
$x + x = x$	A3		
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4	$x + \mathsf{cts}(\delta) = x$	A6ID
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5	$\mathsf{cts}(\delta) \cdot x = \mathsf{cts}(\delta)$	A7ID

Table 1: Axioms of $\mathrm{BPA}_{\mathrm{drt}}^{=}$

We give a structured operational semantics for $\mathrm{BPA}_{\mathrm{drt}}^{=}$. We use rules in the style of Plotkin to define the following relations on $\mathcal{P}$:

action step $\subseteq \mathcal{P} \times \mathsf{A} \times \mathcal{P}$

action termination $\subseteq \mathcal{P} \times \mathsf{A}$

time step $\subseteq \mathcal{P} \times \mathcal{P}$.

We write

$x \xrightarrow{a} x'$ for $(x, a, x') \in$ action step

$x \xrightarrow{a} \surd$ for $(x, a) \in$ action termination

$x \xrightarrow{\sigma} x'$ for $(x, x') \in$ time step.

The rules of Table 2 define these relations by simultaneous induction. So rule (2.1) for example, is read as follows: if x and x' are in the relation $\xrightarrow{a}$, then so are $x + y$ and x' as well as $y + x$ and x'. Rules (1.1) and (4.1) are unconditional. Although there is a negative premise in rule (4.2) these rules define a unique smallest relation. This is not generally the case.

Besides the above mentioned interpretation of the rules, a more operational one can be attached to them. If $x \xrightarrow{a} x'$, this can be read: the process x can perform the action a and then proceed as process x'. The notation $x \xrightarrow{a} \surd$ indicates that the process x can perform an action a and then terminate successfully. Finally, $x \xrightarrow{\sigma} x'$ has as operational meaning that the process x can pass to the next time slice and then proceed as process x'. So rule (2.1) can be read operationally as follows: if process x can perform an action a and then proceed as process x', then the alternative composition of x with some other process y has the choice to perform the action a and after that to proceed as process x'. Note that rules (4.2) and (4.3) have complementary conditions. Together they enforce that the choice between two processes that both can pass to the next time slice is postponed till after the passage to the next time slice. This corresponds to the time determinism property reflected by the axiom DRT1.

The equivalence we use is *bisimulation* on transition systems. It is defined as follows:

Definition 2.1 (bisimulation on transition systems)
A bisimulation relation is a symmetric relation R on process expressions such that:

1. if $R(p, q)$ and $p \xrightarrow{\mu} p'$ for some $\mu \in \mathsf{A}_\sigma$ and process expression p', then there is a process expression q' such that $q \xrightarrow{\mu} q'$ and $R(p', q')$
2. if $R(p, q)$ and $p \xrightarrow{a} \surd$ for some $a \in \mathsf{A}$ then $q \xrightarrow{a} \surd$.

□

Two process expressions p and q are bisimilar, notation $p \leftrightarroweq q$, if there exists a bisimulation relation R relating them. The set of process expressions modulo bisimilarity is a model for $\mathrm{BPA}^{=}_{\mathrm{drt}}$. This kind of model is known in the literature as a term model.

1.1 $\dfrac{}{\mathsf{cts}(a) \xrightarrow{a} \surd}$

2.1 $\dfrac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x', y + x \xrightarrow{a} x'}$ 2.2 $\dfrac{x \xrightarrow{a} \surd}{x + y \xrightarrow{a} \surd, y + x \xrightarrow{a} \surd}$

3.1 $\dfrac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y}$ 3.2 $\dfrac{x \xrightarrow{a} \surd}{x \cdot y \xrightarrow{a} y}$

4.1 $\dfrac{}{\sigma_{\mathsf{rel}}(x) \xrightarrow{\sigma} x'}$ 4.2 $\dfrac{x \xrightarrow{\sigma} x', y \not\xrightarrow{\sigma}}{x + y \xrightarrow{\sigma} x', y + x \xrightarrow{\sigma} x'}$ 4.3 $\dfrac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x + y \xrightarrow{\sigma} x' + y'}$

5.2 $\dfrac{x \xrightarrow{\sigma} x'}{x \cdot y \xrightarrow{\sigma} x' \cdot y}$

Table 2: Operational rules for $\mathrm{BPA}^{=}_{\mathrm{drt}}$

2.2 Recursion

In this subsection we introduce finite linear recursion into the theory of $\mathrm{BPA}^{=}_{\mathrm{drt}}$. The obtained theory will be denoted by $\mathrm{BPA}^{=}_{\mathrm{drt}}\mathrm{Lin}$.

Definition 2.2 (recursive specification)
Let V be a set of variables. A recursive specification $E = E(V)$ in $\mathrm{BPA}^{=}_{\mathrm{drt}}$ is a set of equations

$$E = \{X = E_X(V) | X \in V\}$$

where each $E_X(V)$ is a $\mathrm{BPA}^{=}_{\mathrm{drt}}$ term that only contains variables from V. These equations are called recursion equations. By convention we use capital letters $X, Y, ...$ (possibly decorated with indices) for variables bound in a recursive specification. □

Definition 2.3 (solution of recursive specification)
A solution of a recursive specification $E = E(V)$ in some model of $\mathrm{BPA}^{=}_{\mathrm{drt}}$ is an assignment that attaches to each variable $X \in V$ a process in that model such that the equations of the recursive specification are true statements under this assignment. The process attached to the variable X will be referred to as $\langle X|E\rangle$. If we are interested in a particular variable X we will call $\langle X|E\rangle$ the solution of the recursive specification. □

Definition 2.4 ($\langle t|E\rangle$)
Let E be a recursive specification and let t be a $\mathrm{BPA}^{=}_{\mathrm{drt}}$ term. Then $\langle t|E\rangle$ is the process t with all occurrences of $X \in V$ in t replaced by $\langle X|E\rangle$. □

Definition 2.5 (linear recursive specification)
An recursion equation is said to be linear if the right hand side of the equation has one of the following forms:

- $\sum_{i=1}^{k} \mathsf{cts}(a_i)X_i + \sum_{i=1}^{l} \mathsf{cts}(b_i)$
- $\sum_{i=1}^{k} \mathsf{cts}(a_i)X_i + \sum_{i=1}^{l} \mathsf{cts}(b_i) + \sigma_{\mathsf{rel}}(Y)$

By convention the empty sum is $\mathsf{cts}(\delta)$.

We call a recursive specification linear if all equations are linear. We call a recursive specification finite if it has a finite number of equations. □

The signature of $\mathrm{BPA}_{\mathrm{drt}}^{=}\mathrm{Lin}$ consists of the signature of $\mathrm{BPA}_{\mathrm{drt}}^{=}$ plus a constant $\langle X|E\rangle$ for all finite linear recursive specifications $E = E(V)$ and for all $X \in V$. The axioms of $\mathrm{BPA}_{\mathrm{drt}}^{=}\mathrm{Lin}$ consist of the axioms of $\mathrm{BPA}_{\mathrm{drt}}^{=}$ plus for all recursive specifications $E = E(V)$ and for all $X \in V$ an equation $\langle X|E\rangle = \langle E_X|E\rangle$. We get an operational semantics of $\mathrm{BPA}_{\mathrm{drt}}^{=}\mathrm{Lin}$ by adding the rules given in Table 3 to the rules given in Table 2. We obtain a term model for $\mathrm{BPA}_{\mathrm{drt}}^{=}\mathrm{Lin}$ as in the case of $\mathrm{BPA}_{\mathrm{drt}}^{=}$. In this setting, models for theories involving recursion are considered those models in which the principles RSP and RDP (given below) hold. Note that in the following we refer only to linear recursive specifications.

Definition 2.6 (RDP)
A model is said to satisfy the RDP (Recursive Definition Principle) if every recursive specification E has at least one solution. □

Definition 2.7 (RSP)
A model is said to satisfy RSP (Recursive Specification Principle) if every recursive specification has at most one solution. □

If both RSP and RDP hold, then a recursive specification has a unique solution.

In the untimed case these principles hold for the term model. The proof of this makes use of the bounded-nondeterminism property and of a projection operator. The model we have presented has the bounded-nondeterminism property and a similar projection operator can be given in the timed case (see for example [3]). We conjecture that the model given above is a model of $\mathrm{BPA}_{\mathrm{drt}}^{=}\mathrm{Lin}$.

$$\frac{\langle E_X|E\rangle \xrightarrow{a} x'}{\langle X|E\rangle \xrightarrow{a} x'} \quad \frac{\langle E_X|E\rangle \xrightarrow{a} \surd}{\langle X|E\rangle \xrightarrow{a} \surd} \quad \frac{\langle E_X|E\rangle \xrightarrow{\sigma} x'}{\langle X|E\rangle \xrightarrow{\sigma} x'}$$

Table 3: Additional rules for $\mathrm{BPA}^{=}_{\mathrm{drt}}\mathrm{Lin}$

3 Graph model for discrete time process algebra without recursion

In this section we will present a graph model for $\mathrm{BPA}^{=}_{\mathrm{drt}}$. The elements of the model will be finite directed labelled graphs in which a root node and several successful termination nodes are distinguished. We will use the common notions of "node" and "edge" as well as the notions of "source" and "target" in order to indicate the direction of an edge.

Definition 3.1 (termination node)
A termination node is a node without outgoing edges. □

Definition 3.2 (σ-edges, final edges)
The edges that carry the label σ will be referred to as σ-edges. An edge which has as target a termination node is said to be a final edge. □

The construction of process graphs is after [3] and [5].

Definition 3.3 (process graph)
A process graph is a quadruple $\langle N, E, r, \downarrow\rangle$, where

- N is the set of nodes,
- $E \subseteq (N \setminus \downarrow) \times A_\sigma \times N$ is the set of labelled edges,
- $r \in (N \setminus \downarrow)$ is the root node,
- $\downarrow$ is the set of successful termination nodes (all nodes in $\downarrow$ are termination nodes);

such that the following conditions hold:

1. from any node, there is at most one outgoing σ-edge,
2. a node marked as a successful termination node has no incoming σ-edges,
3. the root is not marked as successful termination node.

If G is a process graph, $N(G)$ will denote the set of nodes of G, $E(G)$ the set of edges of G, $\downarrow(G)$ the set of successful termination nodes, and $r(G)$ the root of G. □

Definition 3.4 (unsuccessful termination)
A node will be called an unsuccessful termination node if it is a termination node and is not in the set of successful termination nodes of the graph. □

The equivalence we use is bisimilarity on graphs which is defined as follows.

Definition 3.5 (bisimulation on graphs)
Let G_1, G_2 be two graphs and R a relation between the nodes of G_1 and the nodes of G_2. Relation R is a bisimulation between G_1 and G_2, notation $R : G_1 \leftrightarroweq G_2$ if it is symmetric and the following conditions hold:

1. the roots of G_1 and G_2 are related;
2. if $u \xrightarrow{\mu} v$, $\mu \in A_\sigma$ is an edge in G_1 and $R(u, u')$, for some node in G_2, then there is an edge $u' \xrightarrow{\mu} v'$ in G_2 and $R(v, v')$;
3. if $R(u, u')$ and $u \downarrow$ then $u' \downarrow$.

□

We will use the root unwinding operation ρ that transforms a graph G to a graph $\rho(G)$.

Definition 3.6 (root unwinding)
Given a graph G, we obtain $\rho(G)$ as follows: add a new node r' to G, add an edge $r' \xrightarrow{\mu} s$ for each edge $r \xrightarrow{\mu} s$ in G ($\mu \in A_\sigma$), and take r' as the root of $\rho(G)$. □

Lemma 3.7 $G \leftrightarroweq \rho(G)$

Proof: Similar to the proof for the untimed case given in [6]. □

We will also use for an arbitrary graph G an operation idn_G, which given two nodes of the graph, u and u', will produce a new node, $\mathsf{idn}_G(u, u')$ such that $\mathsf{idn}_G(u, u') = \mathsf{idn}_G(v, v')$ iff $u = v$ and $u' = v'$.

In the following definitions we suppose, without loss of generality, that the set of nodes of G_1 and G_2 are disjoint. Besides we use r_1 and r_2 to refer to the roots of G_1 and G_2, respectively. The interpretation of the constants and the operators of $\mathrm{BPA}_{\mathrm{drt}}^{=}$ is as follows.

Definition 3.8 (graph model)

1. The process graph $\mathsf{cts}(\delta)_{\mathcal{G}}$ is the process graph with one node that is not marked as a successful termination node u.
2. The process graph $\mathsf{cts}(a)_{\mathcal{G}}$ is the process graph with two nodes, say r and t, where r is the root node and t is marked as a successful termination node, and one edge, viz. $r \xrightarrow{a} t$.
3. Given two process graphs G_1 and G_2, we obtain the process graph $G_1 +_{\mathcal{G}} G_2$ as follows:
   (a) First we obtain a graph H_0 as follows: take the nodes and edges of G_1 and G_2, add the node $\mathsf{idn}_H(r_1, r_2)$, add an edge $\mathsf{idn}_H(r_1, r_2) \xrightarrow{\mu} u$ ($\mu \in A_\sigma$) for each edge $r_1 \xrightarrow{\mu} u$ in G_1 and each edge $r_2 \xrightarrow{\mu} u'$ in G_2, and take $\mathsf{idn}_H(r_1, r_2)$ as the root of H_0. Continue with step (b).
   (b) If the root of H_i, say r', has at most one outgoing σ-edge, then continue with step (c). Otherwise r' has two outgoing σ-edges, say $r' \xrightarrow{\sigma} u$ and $r' \xrightarrow{\sigma} u'$. In this case, we obtain a graph H_{i+1} from H_i as follows: remove the two σ-edges and
      i. if the node $\mathsf{idn}_H(u, u')$ already exists in the graph H_i, then add a σ-edge from the root to $\mathsf{idn}_H(u, u')$;
      ii. if the node $\mathsf{idn}_H(u, u')$ does not exist in the graph H_i, then add it to the graph, add an edge $\mathsf{idn}_H(u, u') \xrightarrow{\mu} v$ ($\mu \in A_\sigma$) for each edge $u \xrightarrow{\mu} v$ and for each edge $u' \xrightarrow{\mu} v$ in H_i, and take $\mathsf{idn}_H(u, u')$ as the root of H_{i+1}.

      In case i continue with step (c) and in case ii repeat step (b).
   (c) Let H be the last graph obtained by performing step (b). We obtain $G_1 +_{\mathcal{G}} G_2$ from H as follows: mark those nodes in H as successful termination nodes that are marked as successful termination nodes in G_1 or G_2, and take $idn_H(r_1, r_2)$ as the root of $G_1 +_{\mathcal{G}} G_2$.
4. Given two process graphs G_1 and G_2, we obtain the process graph $G_1 \cdot_{\mathcal{G}} G_2$ by appending a copy of G_2 to each node of G_1 marked as a successful termination node.
5. Given a process graph G_1, we obtain the process graph $\sigma_{rel_{\mathcal{G}}}(G_1)$ by adding a new root node, say r', and an edge $r' \xrightarrow{\sigma} r1$.

□

Figure 1 indicates how the alternative composition of the graphs corresponding to some recursively defined processes is calculated.

Definition 3.9 ($[\,]_{\leftrightarrow}$)
For $G \in \mathcal{G}$ we will denote by $[G]_{\leftrightarrow}$ the equivalence class of G modulo bisimilarity. □

Bisimulation (Definition 2.1 is a congruence on $\mathcal{G}$ with respect to the operations defined as above. We extend the definition of operations on equivalence classes as usual. We define:

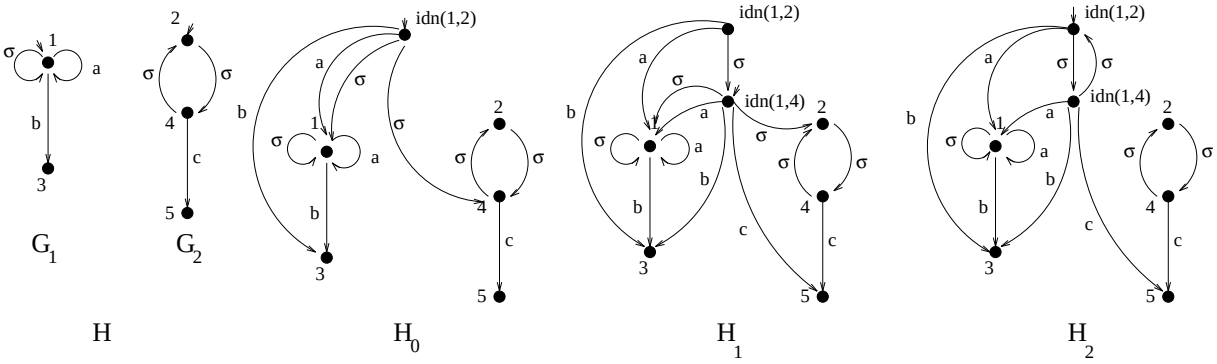

Figure 1: Interpretation of + on process graphs

- $\sigma_{\mathrm{rel}_{\mathcal{G}/\underline{\leftrightarrow}}}([G]\underline{\leftrightarrow}) = [\sigma_{\mathrm{rel}_G}(G)]\underline{\leftrightarrow}$;
- $[G_1]\underline{\leftrightarrow} +_{\mathcal{G}/\underline{\leftrightarrow}} [G_2]\underline{\leftrightarrow} = [G_1 +_{\mathcal{G}} G_2]\underline{\leftrightarrow}$;
- $[G_1]\underline{\leftrightarrow} \cdot_{G/\underline{\leftrightarrow}} [G_2]\underline{\leftrightarrow} = [G_1 \cdot_{\mathcal{G}} G_2]\underline{\leftrightarrow}$.

Lemma 3.10 $(G/\underline{\leftrightarrow}, +_{\mathcal{G}/\underline{\leftrightarrow}}, \cdot_{\mathcal{G}/\underline{\leftrightarrow}}, \sigma_{\mathrm{rel}_{\mathcal{G}/\underline{\leftrightarrow}}}, [\mathsf{cts}(a)_{\mathcal{G}}]\underline{\leftrightarrow}, [\mathsf{cts}(\delta)_{\mathcal{G}}]\underline{\leftrightarrow})$ *satisfies* $BPA^{=}_{\mathrm{drt}}$.

Proof: The equalities that constitute the axioms of $\mathrm{BPA}^{=}_{\mathrm{drt}}$ correspond to bisimulations at the level of graphs. For each axiom, a general construction of a bisimulation can easily be devised. □

4 Timed frames

This section contains a survey of simple timed frame algebra. We refer to [7] for further details.

Timed frames are built from states and transitions between states. The states are obtained by an embedding of naturals in states, and a pairing function on states. Simple timed frames contain two kinds of transitions: action steps and time steps. We consider action steps with a label from a finite set A of *actions*.

The signature of *(simple) timed frames* is as follows:

Sorts:

$\mathbb{N}$	naturals;
$\mathbb{S}$	*states*;
$\mathbb{F}_\mathrm{t}$	*timed frames*;

Constants & Functions:

$0 : \mathbb{N}$	zero;
$S : \mathbb{N} \to \mathbb{N}$	successor;
$\imath_\mathbb{N} : \mathbb{N} \to \mathbb{S}$	embedding of naturals in states;
$\rangle\!\vdash\!\langle : \mathbb{S}^2 \to \mathbb{S}$	pairing of states;
$\emptyset : \mathbb{F}_\mathrm{t}$	*empty timed frame*;
$\imath_\mathbb{S} : \mathbb{S} \to \mathbb{F}_\mathrm{t}$	*embedding of states* in timed frames;
$\xrightarrow{a} : \mathbb{S}^2 \to \mathbb{F}_\mathrm{t}$	*action step construction* (one for each $a \in A$);
$\xrightarrow{\sigma} : \mathbb{S}^2 \to \mathbb{F}_\mathrm{t}$	*time step construction*;
$\oplus : \mathbb{F}_\mathrm{t}^2 \to \mathbb{F}_\mathrm{t}$	*timed frame union.*

Given the signature, (closed) terms are constructed in the usual way. We shall use the meta-variables n and m to stand for arbitrary terms of sort $\mathbb{N}$, the meta-variables s, s' and s'' to stand for arbitrary terms of sort $\mathbb{S}$, and the meta-variables X, Y and Z to stand for arbitrary terms of sort $\mathbb{F}_\mathrm{t}$. We write n instead of $\imath_\mathbb{N}(n)$ or $\imath_\mathbb{S}(\imath_\mathbb{N}(n))$ as well as s instead of $\imath_\mathbb{S}(s)$ when this causes no ambiguity. Terms of the forms $\imath_\mathbb{S}(s)$ and $s \xrightarrow{a} s'$ denote *atomic* timed frames, i.e. timed frames that contain a single state or transition. The constant $\emptyset$ denotes the timed frame that contains neither states nor transitions. The operator $\oplus$ on timed frames gives the union of the states and transitions of its arguments. The pairing function $\rangle\!\vdash\!\langle$ is a simple means to define "fresh" states.[1] The axioms for timed frames are given in Table 4. These axioms characterise frames as

(FA1)	$X \oplus Y$	$=$	$Y \oplus X$
(FA2)	$X \oplus (Y \oplus Z)$	$=$	$(X \oplus Y) \oplus Z$
(FA3)	$X \oplus X$	$=$	X
(FA4)	$X \oplus \emptyset$	$=$	X
(FA5)	$s \oplus (s \xrightarrow{a} s')$	$=$	$s \xrightarrow{a} s'$
(FA6)	$s' \oplus (s \xrightarrow{a} s')$	$=$	$s \xrightarrow{a} s'$
(TFA1)	$s \oplus (s \xrightarrow{\sigma} s')$	$=$	$s \xrightarrow{\sigma} s'$
(TFA2)	$s' \oplus (s \xrightarrow{\sigma} s')$	$=$	$s \xrightarrow{\sigma} s'$

Table 4: Axioms for timed frames.

objects consisting of a finite set of states and a finite set of transitions (axioms (FA1)–(FA4)). In addition, frames are identified if they are the same after addition of the states occurring in the transitions to the set of states (axioms (FA5), (FA6), (TFA1) and (TFA2)).

We define *iterated* frame union by

$$\bigoplus_{i=n}^{k} X_i = \begin{cases} \emptyset & \text{if } k < n, \\ X_n \oplus \bigoplus_{i=n+1}^{k} X_i & \text{otherwise.} \end{cases}$$

[1] In [8], where time-free frames were introduced, the pairing function is used to define a *frame product* function.

Every frame has a finite number of states and transitions. In [8], frame polynomials are introduced to deal with the countably infinite case as well. This paper focuses on timed frames corresponding to *regular* discrete time processes. Therefore only frames with a finite number of states and transitions are considered.

In order to investigate the connection with discrete time process algebra, we introduce in Definition 4.3 a special kind of bisimulation, called σ-bisimulation. That definition and subsequent ones need some conditions that are related to the transitions contained in a given frame. These frame conditions are as follows.

Definition 4.1 (frame conditions)

$$[s \xrightarrow{a} s']_F = \begin{cases} \mathsf{t} & \text{if } (s \xrightarrow{a} s') \oplus F = F \\ \mathsf{f} & \text{otherwise} \end{cases}$$

$$[s \xrightarrow{\sigma} s']_F = \begin{cases} \mathsf{t} & \text{if } (s \xrightarrow{\sigma} s') \oplus F = F \\ \mathsf{f} & \text{otherwise} \end{cases}$$

$$[s \rightarrow s']_F = \begin{cases} \mathsf{t} & \text{if } [s \xrightarrow{a} s']_F = \mathsf{t} \text{ for some } a \text{ or } [s \xrightarrow{\sigma} s']_F = \mathsf{t} \\ \mathsf{f} & \text{otherwise} \end{cases}$$

$$[s \rightarrow_S^* s']_F = \begin{cases} \mathsf{t} & \text{if } [s \rightarrow s']_F = \mathsf{t} \text{ or} \\ & \quad [s \rightarrow s'']_F = \mathsf{t} \text{ and } [s'' \rightarrow_S^* s']_F = \mathsf{t} \text{ for some } s'' \in S \\ \mathsf{f} & \text{otherwise} \end{cases}$$

□

In the sequel, we will write $[s \xrightarrow{a} s']_F$ instead of $[s \xrightarrow{a} s']_F = \mathsf{t}$, $[s \xrightarrow{\sigma} s']_F$ instead of $[s \xrightarrow{\sigma} s']_F = \mathsf{t}$, etc. when this causes no ambiguity. We write $|F|$ for $\{s \in \mathcal{S} \mid \imath_{\mathcal{S}}(s) \oplus F = F\}$. For $s' \in |F|$, we write $[\not\xrightarrow{\sigma} s']_F$ to indicate that there exists no $s \in |F|$ such that $[s \xrightarrow{\sigma} s']_F$ and $[s \not\xrightarrow{a}]_F$ to indicate that there exists no $s'' \in |F|$ such that $[s \xrightarrow{a} s'']_F$.

Below bisimulation and σ-bisimulation are defined as equivalences on pointed frames, i.e. frames equipped with a root marker and a termination marker. For further explanation of σ-bisimulation we refer to [7]. Pointed frames, which are closely related to transition systems, are defined first.

Definition 4.2 (pointed timed frame)
A *pointed* timed frame is a triple (F, p, q) where F is a timed frame and $p, q \in |F|$. □

Definition 4.3 (σ-bisimulation)
Let F and F' be timed frames, and let $p, q \in |F|$ and $p', q' \in |F'|$. The pointed timed frames (F, p, q) and (F', p', q') are *σ-bisimilar*, written $(F, p, q) \overset{\sigma}{\leftrightarrows} (F', p', q')$, if there exists a relation R on $\mathcal{P}(|F|) \times \mathcal{P}(|F'|)$ such that:

1. $R(\{p\},\{p'\})$;
2. if $R(S,T)$ and $[s \xrightarrow{a} s']_F$ for some $s \in S$ and $s' \in |F| \backslash \{q\}$, then $[t \xrightarrow{a} t']_{F'}$ and $R(\{s'\},\{t'\})$ for some $t \in T$ and $t' \in |F'| \setminus \{q'\}$;

2^c. rule 2 vice versa;

3. if $R(S,T)$ and $[s \xrightarrow{a} q]_F$ for some $s \in S$, then $[t \xrightarrow{a} q']_{F'}$ for some $t \in T$;

3^c. rule 3 vice versa;

4. if $R(S,T)$, then $R(S',T')$ where $S' = \{s' \in |F| \setminus \{q\} \mid \exists s \in S \cdot [s \xrightarrow{\sigma} s']_F\}$ and $T' = \{t' \in |F'| \setminus \{q'\} \mid \exists t \in T \cdot [t \xrightarrow{\sigma} t']_{F'}\}$;
5. if $R(S,T)$ and $[s \xrightarrow{\sigma} s']_F$ for some $s \in S$ and $s' \in |F|$, then $[t \to t']_{F'}$ for some $t \in T$ and $t' \in |F'|$;

5^c. rule 5 vice versa.

(F,p,q) and (F',p',q') are *bisimilar*, written $(F,p,q) \leftrightarrows (F',p',q')$, if there exists a relation R that satisfies, in addition to the above-mentioned conditions, the following one:

6. if $R(S,T)$, then $card(S) = card(T) \leq 1$.

□

σ-bisimulation is an equivalence relation on frames. We will write $R : (F,p,q) \overset{\sigma}{\leftrightarrows} (F',p',q')$ to indicate that R is a σ-bisimulation relation between the frames (F,p,q) and (F',p',q').

For $S \subseteq |F|$, we write $\sigma_{(F,p,q)}(S)$ for $\{s' \in |F| \setminus \{q\} \mid \exists s \in S \cdot [s \xrightarrow{\sigma} s']_F\}$.

Lemma 4.4 *If (F,p,q) is a pointed frame then there is an auto σ-bisimulation on (F,p,q) which relates only sets of states that are reachable from p.*

Proof: Let $F^2 = \{(M,M) | M \in \mathcal{P}(|F|)\}$. For every $B \subseteq F^2$ define $N(B)$ as the smallest subset of F^2 with the following properties:

1. if $(M,M) \in B, s \in M$ and $[s \xrightarrow{a} r]_F$ then $(\{r\},\{r\}) \in N(B)$;
2. if $(M,M) \in B$ then $(\sigma_{(F,p,q)}(M), \sigma_{(F,p,q)}(M)) \in N(B)$.

It is immediate that $B \subseteq F^2$ implies $N(B) \subseteq F^2$. Let $B_0 = \{(\{p\},\{p\})\}$ and $R = \bigcup_{n \in \mathbb{N}} N^n(B_0)$, where $N^0(B0) = B_0$ and $N^{n+1}(B0) = N(N^n(B0))$. It can be easily shown by induction on n that $s \in M$ and $(M,M) \in N^n(B_0)$ imply s is reachable from p with n transitions. We now show that R is an auto σ-bisimulation on (F,p,q) by checking each of the conditions for σ-bisimulation.

1. $R(\{p\},\{p\})$, because $B_0 \in R$.
2. Suppose that $R(S,S)$, $[s \xrightarrow{a} r]_F$ for some $s \in S$ and $r \in |F| \setminus \{q\}$. Then $(S,S) \in N^k(B_0)$ for some $k \in \mathbb{N}$ and $(\{r\},\{r\}) \in N^{k+1}(B_0)$, thus $R(\{r\},\{r\})$.
3. Immediate.
4. Suppose $R(S,S)$. Then $(S,S) \in N^k(B_0)$ for some $k \in \mathbb{N}$ and $(\sigma_{(F,p,q)}(S), \sigma_{(F,p,q)}(S)) \in N^{k+1}(B_0)$, thus $R(\sigma_{(F,p,q)}(S), \sigma_{(F,p,q)}(S))$.
5. Immediate.

□

Definition 4.5 (frame isomorphism)
Two pointed frames (F,p,q) and (F',p',q') are said to be isomorphic if there is a bijection between states which preserves the transitions, the starting state and the termination state, i.e. there is a bijection $f : |F| \to |F'|$ such that $f(p) = p'$, $f(q) = q'$ and $[s \xrightarrow{\mu} t]_F$ iff $[f(s) \xrightarrow{\mu} f(t)]_{F'}$, for any $\mu \in \mathsf{A}_\sigma$. □

Lemma 4.6 *Two isomorphic frames are σ-bisimilar.*

Proof: Suppose that f is an isomorphism between(F,p,q) and (F',p',q'). Let R be an auto σ-bisimulation on (F,p,q), and for $S \subseteq |F|$ denote by $f(S)$ the set $\{f(s) | s \in S\}$. Then $R' = \{(S, f(T)) | R(S,T)\}$ is a σ-bisimulation between (F,p,q) and (F',p',q'). □

We now define time determinism and time persistency for frames, because together they characterise the kind of frames that corresponds to the timed transition systems that underlie the model of discrete time process algebra with relative timing presented in [3]. Frames of this kind are called proper timed frames.

Definition 4.7 (σ-deterministic, σ-persistent, proper timed frames)
A timed frame F is *σ-deterministic* if it satisfies:

if $[s \xrightarrow{\sigma} t]_F$ and $[s \xrightarrow{\sigma} t']_F$ for some $s,t,t' \in |F|$, then $t = t'$.

A timed frame F is *σ-persistent* if it satisfies:

if $[s \xrightarrow{a} t]_F$ and $[s \xrightarrow{\sigma} t']_F$ for some $s,t,t' \in |F|$, then $[t' \to t'']_F$ for some $t'' \in |F|$.

A timed frame F is *proper* if it is σ-deterministic and σ-persistent. $\Box$

In [3], discrete time process algebra with relative timing is based on transition systems corresponding to pointed frames (F,p,q) where F is proper and q has no incoming time steps (i.e. $[\stackrel{\sigma}{\not\rightarrow} q]_F$). For pointed frames satisfying these conditions, the definition of bisimulation given here is equivalent to the one given in that paper.

According to the following three lemmas, every pointed frame is σ-bisimilar to one of the pointed frames that correspond to the transition systems that underlie the model of discrete time process algebra with relative timing presented in [3].

Lemma 4.8 *Every pointed timed frame (F,p,q) is σ-bisimilar to a pointed timed frame (F',p,q) where $[\stackrel{\sigma}{\not\rightarrow} q]_{F'}$.*

Proof: Add a new state, q', to the frame and replace q by q' in all its incoming time steps. The frame obtained is σ-bisimilar with the original one. $\Box$

Lemma 4.9 *Every pointed timed frame (F,p,q) where $[\stackrel{\sigma}{\not\rightarrow} q]_F$ is σ-bisimilar to a pointed timed frame (F',p',q') where F' is σ-deterministic and $[\stackrel{\sigma}{\not\rightarrow} q]_{F'}$.*

Proof: Let $en : \mathcal{P}(|\mathcal{F}|) \cup \{q\} \rightarrow \mathbb{N}$ be an enumeration of the elements of the power set of $|F|$. Construct a frame F' as follows:

1. for each $S \in \mathcal{P}(|F|)$, there is a state $en(S)$ in the frame;
2. for each $S \in \mathcal{P}(|F|)$, if $[s \stackrel{a}{\rightarrow} t]_F$ for some $s \in S$, then the action step $en(S) \stackrel{a}{\rightarrow} en(\{t\})$ is in the frame;
3. for each $S \in \mathcal{P}(|F|)$, if $\sigma_{(F,p,q)}(S) \neq \emptyset$, then the time step $en(S) \stackrel{\sigma}{\rightarrow} en(\sigma_{(F,p,q)}(S))$ is in the frame;
4. for each $S \in \mathcal{P}(|F|)$, if $\sigma_{(F,p,q)}(S) = \emptyset$ and $[s \stackrel{\sigma}{\rightarrow} q]_F$ for some $s \in S$ then $en(S) \stackrel{\sigma}{\rightarrow} en\{q\}$ is in the frame.

F' is clearly σ-deterministic. Let $p' = en(\{p\})$ and $q' = en(\{q\})$. Let R be an auto σ-bisimulation on (F,p,q). Then R' defined by $R'(\emptyset,\emptyset)$ and $R'(S,en(S))$ iff $R(S,S)$ is a σ-bisimulation between (F,p,q) and (F',p',q'). $\Box$

Lemma 4.10 *Every pointed timed frame (F,p,q) is σ-bisimilar to a pointed timed frame (F',p,q) where F' is σ-persistent.*

Proof: Construct F' from F by removing all the final time steps provided that they have outgoing action steps from their source. The frame F' is obviously proper. Let $R : (F,p,q) \stackrel{\sigma}{\leftrightarrows} (F,p,q)$ be an auto σ-bisimulation. A σ-bisimulation $R : (F,p,q) \stackrel{\sigma}{\leftrightarrows} (F',p',q')$ is obtained as follows: $R'(S,T')$ iff $R(S,T)$ and T' is T without the targets of the removed time steps. □

We will use the following fact:

Remark 4.11 Let (F,p,q) be a proper frame and let (F',p',q') be a frame that is obtained by applying the constructions (either one of them or both) given in the proofs of the Lemmas 4.9 and 4.10. Then the part of F reachable from p is isomorphic with the part of F' reachable from p'. □

According to the following lemma, $\stackrel{\sigma}{\leftrightarrows}$ and $\leftrightarrows$ coincide for the pointed frames that correspond to the transition systems that underlie the model of discrete time process algebra with relative timing presented in [3].

Lemma 4.12 *For any two proper timed frames F and F', $p,q \in |F|$ and $p',q' \in |F'|$, $(F,p,q) \stackrel{\sigma}{\leftrightarrows} (F',p',q')$ iff $(F,p,q) \leftrightarrows (F',p',q')$.*

Proof: The proof is analogous to the proof for Lemma 3.7 from [7]. □

5 Frame model for discrete time process algebra without recursion

In this section, we introduces a timed frame model for $\mathrm{BPA}^{=}_{\mathrm{drt}}$-ID without recursion. We extend this frame model for $\mathrm{BPA}^{=}_{\mathrm{drt}}$-ID with finite linear recursion in the next section.

We use timed frame algebra to give an interpretation of the constants and operators of $\mathrm{BPA}^{=}_{\mathrm{drt}}$-ID on frames. This extends to a model of $\mathrm{BPA}^{=}_{\mathrm{drt}}$-ID since σ-bisimulation is a congruence with respect to the interpretation of the operators.

We first define some useful auxiliary operations on frames.

To begin with, we shall use the extension of the successor function S to states and frames. This extension will be used to make the set of states of one frame disjunct from the set of states of another frame. It is straightforward to define the extension:

$$\begin{array}{rcl}
S(\imath_{\mathbb{N}}(n)) & = & \imath_{\mathbb{N}}(S(n)) \\
S(s \mathrel{)\!\!-\!\!\!<} s') & = & S(s) \mathrel{)\!\!-\!\!\!<} S(s') \\
S(\emptyset) & = & \emptyset \\
S(\imath_{\mathbb{S}}(s)) & = & \imath_{\mathbb{S}}(S(s)) \\
S(s \xrightarrow{a} t) & = & S(s) \xrightarrow{a} S(t) \\
S(s \xrightarrow{\sigma} t) & = & S(s) \xrightarrow{\sigma} S(t) \\
S(X \oplus Y) & = & S(X) \oplus S(Y)
\end{array}$$

We also simply write $S^n(E)$ for the nth successor of E, where E is a term of sort $\mathbb{N}$, $\mathbb{S}$ or $\mathbb{F}_{\mathrm{t}}$. This notation can be defined as follows:

$$\begin{array}{rcl}
S^0(E) & = & E \\
S^{n+1}(E) & = & S(S^n(E))
\end{array}$$

Furthermore, we shall use operations $\rho_{\mathsf{src}}, \rho^{\circ}_{\mathsf{tgt}}, \rho^{\bullet}_{\mathsf{tgt}} : \mathbb{S} \times \mathbb{S} \times \mathbb{F}_{\mathrm{t}} \to \mathbb{F}_{\mathrm{t}}$ to replace a state in each of its outgoing transitions, in each its incoming action steps and in each its incoming time steps, respectively. These replacements operations will be used to identify the root or termination state of one frame with the root or termination state of another frame. It is rather straightforward to define the replacement operations:

$$\begin{array}{rcll}
\rho_{\mathsf{src}(s/s')}(\emptyset) & = & \emptyset & \\
\rho_{\mathsf{src}(s/s')}(s'') & = & s'' & \\
\rho_{\mathsf{src}(s/s')}(s \xrightarrow{a} t) & = & s' \xrightarrow{a} t \oplus s & \\
\rho_{\mathsf{src}(s/s')}(s'' \xrightarrow{a} t) & = & s'' \xrightarrow{a} t & \text{if } s \neq s'' \\
\rho_{\mathsf{src}(s/s')}(s \xrightarrow{\sigma} t) & = & s' \xrightarrow{\sigma} t \oplus s & \\
\rho_{\mathsf{src}(s/s')}(s'' \xrightarrow{\sigma} t) & = & s'' \xrightarrow{\sigma} t & \text{if } s \neq s'' \\
\rho_{\mathsf{src}(s/s')}(X \oplus Y) & = & \rho_{\mathsf{src}(s/s')}(X) \oplus \rho_{\mathsf{src}(s/s')}(Y) &
\end{array}$$

$$\begin{array}{rcll}
\rho^{\circ}_{\mathsf{tgt}(t/t')}(\emptyset) & = & \emptyset & \\
\rho^{\circ}_{\mathsf{tgt}(t/t')}(s'') & = & s'' & \\
\rho^{\circ}_{\mathsf{tgt}(t/t')}(s \xrightarrow{a} t) & = & s \xrightarrow{a} t' \oplus t & \\
\rho^{\circ}_{\mathsf{tgt}(t/t')}(s \xrightarrow{a} t'') & = & s \xrightarrow{a} t'' & \text{if } t \neq t'' \\
\rho^{\circ}_{\mathsf{tgt}(t/t')}(s \xrightarrow{\sigma} t'') & = & s \xrightarrow{\sigma} t'' & \\
\rho^{\circ}_{\mathsf{tgt}(t/t')}(X \oplus Y) & = & \rho^{\circ}_{\mathsf{tgt}(t/t')}(X) \oplus \rho^{\circ}_{\mathsf{tgt}(t/t')}(Y) &
\end{array}$$

$$\begin{array}{rcll}
\rho^{\bullet}_{\mathsf{tgt}(t/t')}(\emptyset) & = & \emptyset & \\
\rho^{\bullet}_{\mathsf{tgt}(t/t')}(s'') & = & s'' & \\
\rho^{\bullet}_{\mathsf{tgt}(t/t')}(s \xrightarrow{a} t'') & = & s \xrightarrow{a} t'' & \\
\rho^{\bullet}_{\mathsf{tgt}(t/t')}(s \xrightarrow{\sigma} t) & = & s \xrightarrow{\sigma} t' \oplus t & \\
\rho^{\bullet}_{\mathsf{tgt}(t/t')}(s \xrightarrow{\sigma} t'') & = & s \xrightarrow{\sigma} t'' & \text{if } t' \neq t'' \\
\rho^{\bullet}_{\mathsf{tgt}(t/t')}(X \oplus Y) & = & \rho^{\bullet}_{\mathsf{tgt}(t/t')}(X) \oplus \rho^{\bullet}_{\mathsf{tgt}(t/t')}(Y) &
\end{array}$$

Note that, even if there will be no incoming or outgoing transitions left for the state to be replaced, these operations do not remove a state from a frame.

Root unwinding can simply be defined in terms of the other operations:

$$\upsilon(X,r) \;=\; \rho_{\mathsf{src}(S(r)/0)}(S(X)) \oplus S(X)$$

With these auxiliary operations, it is now easy to give the interpretation of the constants and operators of $\mathrm{BPA}_{\mathrm{drt}}^{=}$ on pointed timed frames. The interpretations are given by the definitions in Table 5. They are denoted by the constant or operator decorated with the subscript $\mathcal{F}$.

$$\begin{array}{rcl}
\mathsf{cts}(a)_{\mathcal{F}} & = & (0 \stackrel{a}{\longrightarrow} 1,0,1) \\
\mathsf{cts}(\delta)_{\mathcal{F}} & = & (0 \stackrel{\sigma}{\longrightarrow} 1,0,1) \\
 & & \\
\sigma_{\mathsf{rel}\,\mathcal{F}}((X,r,t)) & = & (0 \stackrel{\sigma}{\longrightarrow} S(r) \oplus S(X), 0, S(t)) \\
(X,r,t) \cdot_{\mathcal{F}} (X',r',t') & = & (\rho^{\circ}_{\mathsf{tgt}(t/S^n(r'))}(\rho^{\bullet}_{\mathsf{tgt}(t/S^n(t'))}(X)) \oplus S^n(X')), r, S^n(t')) \\
 & & \text{where } n = S(\max(|X|)) \\
(X,r,t) +_{\mathcal{F}} (X',r',t') & = & (\rho^{\circ}_{\mathsf{tgt}(S(t)/S^{n+1}(t'))}(\rho^{\bullet}_{\mathsf{tgt}(S(t)/S^{n+1}(t'))}(X_{ru})) \oplus X'_{ru}, 0, S^{n+1}(t')) \\
 & & \text{where } \begin{array}{l} X_{ru} = \upsilon(X,r), \\ n \;\; = S(\max(|X_{ru}|)), \\ X'_{ru} = \upsilon(S^n(X'), S^n(r')) \end{array}
\end{array}$$

Table 5: Interpretation of constants and operators

Let $\mathcal{F}$ be the set of pointed timed frames (F,p,q) that satisfy:

1. $[p \rightarrow^*_{|F|} s]_F \Rightarrow (s=q) \vee [s \longrightarrow s']_F$ for some $s' \in |F|$;
2. $\neg[q \rightarrow s']_F$ for any $s' \in |F|$.

For $(F,p,q) \in \mathcal{F}$ we will denote by $[(F,p,q)]_{\underline{\leftrightarrow}^{\sigma}}$ the equivalence class of (F,p,q) on $\mathcal{F}$ modulo σ-bisimilarity. We will prove that $(\mathcal{F}/\underline{\leftrightarrow}^{\sigma}, +_{\mathcal{F}/\underline{\leftrightarrow}^{\sigma}}, \cdot_{\mathcal{F}/\underline{\leftrightarrow}^{\sigma}}, [\mathsf{cts}(a)_{\mathcal{F}}]_{\underline{\leftrightarrow}^{\sigma}}, [\mathsf{cts}(\delta)_{\mathcal{F}}]_{\underline{\leftrightarrow}^{\sigma}})$ satisfies $\mathrm{BPA}_{\mathrm{drt}}^{=}$ by showing it is isomorphic with the graph model described in Section 3.

First note that the frames and the graphs are closely related. This is reflected by the following transformation:

Definition 5.1 (underlying graph)
To a given frame a finite directed labelled graph can be attached in a natural way. For each state of the frame there is a node in the graph. For each transition in the frame there is an edge in the graph from the node corresponding to the source of the transition to the node corresponding to the target of the transition. If the transition is an action step then the edge is labelled with the action concerned, otherwise the edge is labelled with σ. The graph obtained is unique modulo isomorphism and it will be referred to as the graph underlying the original frame. We use as

notation for the underlying graph of the frame F, $\mathsf{ugr}(F)$. To avoid a heavy notation we denote the node corresponding to a state s also by s, since it will always be clear from the context if we refer the state or to the corresponding node. We can extend now the definition to pointed frames. The graph $\mathsf{ugr}(F,p,q)$ is the graph underlying F in which we distinguish a root, p, and a successful termination node, q. □

The above transformation is not enough to relate $\mathcal{G}$ and $\mathcal{F}$ in a satisfactory way. This is indicated in the following remark.

Between the graph model and the frame model there are two major differences. Firstly, the latter admits time nondeterminism, i.e. there may be more than one time step outgoing from a certain state and, as a consequence, the underlying graph of a frame is not necessarily a process graph. Secondly, although in both models process behaviours have similar representations, the role of the (unsuccessful termination) nodes of a process graph that represent deadlock is taken by the states of a frame which have an outgoing final time step. This is a reasonable way to represent deadlock because of two reasons: it allows for extensions to frame models for process algebras which include $\dot{\delta}$, and it yields consistency between the interpretation of processes as frames and the process extraction operator defined in [7].

We will devise a transformation gr2fr between the set of process graphs $\mathcal{G}$ and the set of pointed frames $\mathcal{F}$, as well as an inverse transformation fr2gr from $\mathcal{F}$ to $\mathcal{G}$. We further prove that the transformations induced on equivalence classes of G and of $\mathcal{F}$, respectively, are inverse functions of each other. Then by proving that the function induced by gr2fr to equivalence classes commutes with the operations we actually obtain an isomorphism between the graph and the frame model.

Definition 5.2 ($\mathsf{gr2fr}_g$)
Let $G=(N,E,r,\downarrow)$ be a graph in $\mathcal{G}$. On G perform the following transformations:

(a) add an outgoing σ-edge (to a new node) to all unsuccessful termination nodes;
(b) if there is no termination node add a new node q to the graph; otherwise identify all termination nodes and let q be the node obtained.

Let G' be the graph obtained after this transformation. Consider $g : N(G') \to \mathbb{N}$ an arbitrary enumeration of the nodes of G'. We define $\mathsf{gr2fr}_g(G)$ to be the pointed frame $(F, g(r(G)), g(q)$ where $|F| = \{g(u) \mid u \in N(G) \setminus {\downarrow}(G)\} \cup \{g(q)\}$ and $[g(u) \xrightarrow{\mu} g(u')]_{\mathsf{gr2fr}_g(G)}$ iff $u \xrightarrow{\mu} u'$ $(\mu \in \mathsf{A}_\sigma)$ is an edge of G'. □

Definition 5.3 (fr2gr)
Let $(F,p,q) \in \mathcal{F}$ be a pointed timed frame. On (F,p,q)

(a) apply the construction in the proof of Lemma 4.8 (the frame obtained has the property that $[\xrightarrow{\sigma}\!\!\!\!\!/\ q]_F$ and is σ-bisimilar to (F,p,q));

(b) apply the construction in the proof of Lemma 4.9 (the frame obtained is σ-deterministic and is σ-bisimilar to (F,p,q));

(c) apply the construction in the proof of Lemma 4.10 (the frame obtained is σ-persistent and is σ-bisimilar to (F,p,q));

(d) remove all remaining final time steps (the frame obtained has as underlying graph a process graph).

Let (F',p',q') be the frame obtained in this way. We define $\mathsf{fr2gr}((F,p,q))$ to be the graph $\mathsf{ugr}(F',p',q')$. □

Lemma 5.4 *If $G \in \mathcal{G}$ and $\mathsf{gr2fr}_g(G) = (F,p,q)$ then F is a proper frame.*

Proof: Straughtforward from the definition of process graphs. □

Lemma 5.5 $\mathsf{gr2fr}_g(G)$ *does not depend on g, i.e.* $\mathsf{gr2fr}_g(G) \leftrightarrows^{\sigma} \mathsf{gr2fr}_{g'}(G)$ *for any g and g'.*

Proof: Let G' be the graph obtain after steps (a) and (b) from the definition of $\mathsf{gr2fr}_g$. If R is an auto-bisimulation of G' then the relation defined by $R' = \{(\{g(u)\},\{g'(u')\}) | (u,u') \in R\} \cup \{(\emptyset,\emptyset)\}$ is a bisimulation between $\mathsf{gr2fr}_g(G)$ and $\mathsf{gr2fr}_{g'}(G)$. Then from the previous lemma and 4.12, $\mathsf{gr2fr}_g(G) \leftrightarrows^{\sigma} \mathsf{gr2fr}_{g'}(G)$. □

Remark 5.6 If (F,p,q) is an arbitrary pointed frame and (F',p',q') is the frame obtained from (F,p,q) after the first three steps of $\mathsf{fr2gr}$ then F' is proper and $(F,p,q) \leftrightarrows^{\sigma} (F'p',q')$. □

Lemma 5.7 *Let $G_1 \leftrightarrows G_2$ and let g_1 and g_2 be the enumerations used in* $\mathsf{gr2fr}_{g_1}(G_1)$ *and* $\mathsf{gr2fr}_{g_2}(G_2)$. *Then* $\mathsf{gr2fr}_{g_1}(G_1) \leftrightarrows^{\sigma} \mathsf{gr2fr}_{g_2}(G_2)$.

Proof: For $i \in \{1,2\}$, let q_i be the node which is obtained in step (b) of $\mathsf{gr2fr}_{g_i}$ when it is applied to the graph G_i. Suppose $R : G_1 \leftrightarrows G_2$, $\mathsf{gr2fr}_{g_1}(G_1) = (F_1, g_1(r(G_1)), g_1(q_1))$ and $\mathsf{gr2fr}_{g_2}(G_2) = (F_2, g_2(r(G_2)), g_2(q_2))$. Then taking into account Lemma 5.4 it is straightforward to show that if $R' = \{(\{g_1(s)\},\{g_2(s')\}) | (s,s') \in R, s \notin \downarrow(G_1), s' \notin \downarrow(G_2)\} \cup \{(\emptyset,\emptyset)\}$, then $R' : (F_1,p_1,q_1) \leftrightarrows^{\sigma} (F_2,p_2,q_2)$. □

Lemma 5.8 *If $(F,p,q) \leftrightarrows^{\sigma} (F',p',q')$ then $\mathsf{fr2gr}((F,p,q)) \leftrightarrows \mathsf{fr2gr}((F',p',q'))$.*

Proof: Suppose $\mathsf{fr2gr}((F,p,q)) = G_1$ and $\mathsf{fr2gr}((F',p',q')) = G_2$. Let (F_1,p_1,q_1) and (F_1',p_1',q_1') be the frames obtained from (F,p,q) and (F',p',q'), respectively, after applying steps (a), (b)

and (c) from the definition of $\mathsf{fr2gr}$. Then it follows from the assumptions and Remark 5.6 that $(F_1,p_1,q_1) \overset{\sigma}{\leftrightarrows} (F,p,q) \overset{\sigma}{\leftrightarrows} (F',p',q') \overset{\sigma}{\leftrightarrows} (F_1',p_1',q_1')$.

By Lemma 4.12 and once more 5.6, there is $R : (F_1,p_1,q_1) \leftrightarrows (F_1',p_1',q_1')$. The relation R' defined by $R'(q_1,q_1')$ and $R'(u,v)$ iff $R(\{u\},\{v\})$ is a bisimulation between $\mathsf{ugr}(F_1,p_1,q_1)$ and $\mathsf{ugr}(F_1',p_1',q_1')$.

Now, the graphs $\mathsf{fr2gr}((F,p,q))$ and $\mathsf{fr2gr}((F',p',q'))$ can be obtained from $\mathsf{ugr}((F_1,p_1,q_1))$ and $\mathsf{ugr}((F_1',p_1',q_1'))$, respectively, by removing the remaining final σ-edges. It follows that R' is a bisimulation between $\mathsf{fr2gr}((F,p,q))$ and $\mathsf{fr2gr}((F',p',q'))$ as well. □

Lemma 5.9 *If $G \in \mathcal{G}$ then $\mathsf{fr2gr}(\mathsf{gr2fr}_g(G)) \leftrightarrows G$.*

Proof: Suppose that $\mathsf{gr2fr}(G) = (F, g(r(G)), g(q))$. In the case that G has no termination node, $\mathsf{ugr}(F, g(r(G)), g(q))$ is the graph G in which each node u is renamed to $g(u)$ and a new node $g(q)$ is added. Obviously $\mathsf{ugr}(F, g(r(G)), g(q))$ is bisimilar to G. Applying $\mathsf{fr2gr}$ to $(F, g(r(G)), g(q))$ will have no effect on $(F, g(r), g(q))$ thus $\mathsf{fr2gr}(\mathsf{gr2fr}_g(G)) \overset{\sigma}{\leftrightarrows} G$. In the other case ($G$ has at least one termination node), the graph $\mathsf{ugr}(F, g(r(G)), g(q))$ is the graph G in which a σ-edge is added to all unsuccessful termination nodes, all termination nodes are identified and each node u is renamed to $g(u)$. Because $(F, g(r(G)), g(q))$ is a proper frame (Remark 5.4), $\mathsf{fr2gr}((F, g(r(G)), g(q)))$ is obtained by applying steps (a) and (d) from the definition of $\mathsf{fr2gr}$ (Remark 4.11). This will have as effect the removal of the time steps corresponding to the σ-edges added in step (a) of $\mathsf{gr2fr}_g$. The graph which is obtained is the graph G in which each node u is renamed to $g(u)$ and all successful termination nodes are identified. The equivalence $\mathsf{fr2gr}(\mathsf{gr2fr}_g(G) \leftrightarrows G$ is obvious. □

Lemma 5.10 *If $(F,p,q) \in \mathcal{F}$ then $\mathsf{gr2fr}_g(\mathsf{fr2gr}((F,p,q))) \overset{\sigma}{\leftrightarrows} (F,p,q)$.*

Proof: Let (F',p',q') be the frame obtained from (F,p,q) after applying steps (a)-(c) of $\mathsf{fr2gr}$. According to 5.6, $(F,p,q) \overset{\sigma}{\leftrightarrows} (F',p',q')$. Step (d) of $\mathsf{fr2gr}$ and (a) of $\mathsf{gr2fr}_g$ are complementary in the sense that the time steps which are removed in step (d) of $\mathsf{fr2gr}$ are replaced by the time steps corresponding to the σ-edges added in step (a) of $\mathsf{gr2fr}_g$. The graph underlying (F',p',q') is isomorphic with the graph underlying $\mathsf{gr2fr}_g(\mathsf{fr2gr}((F,p,q)))$. □

We define the transformations induced by $\mathsf{gr2fr}_g$ and $\mathsf{fr2gr}$ on equivalence classes and prove that these transformations are functions. Further we prove that the transformation induced by $\mathsf{gr2fr}$ is a homomorphism with respect to the operations on $\mathcal{G}/\leftrightarrows$.

Define $\mathsf{gr2fr}([G]\leftrightarrows) = [\mathsf{gr2fr}_g(G)]_{\overset{\sigma}{\leftrightarrows}}$ for an arbitrary g. Note that the definition does not depend on g (from 5.5) and the transformation which is obtained is a function between equivalence classes (from 5.7). Overloading $\mathsf{fr2gr}$ we define $\mathsf{fr2gr}([(F,p,q)]_{\overset{\sigma}{\leftrightarrows}}) = [\mathsf{fr2gr}((F,p,q)]\leftrightarrows$.

Lemma 5.11 *σ-bisimilarity is a congruence on $(\mathcal{F}, +_{\mathcal{F}}, \cdot_{\mathcal{F}}, \mathsf{cts}(a)_{\mathcal{F}}, \mathsf{cts}(\delta)_{\mathcal{F}})$.*

Proof: We prove that the operations on $\mathcal{F}$ respect σ-bisimilarity. Let (F_1, p_1, q_1), (F_1', p_1', q_1'), (F_2, p_2, q_2), (F_2', p_2', q_2') be pointed timed frames in $\mathcal{F}$, such that $R_1 : (F_1, p_1, q_1) \stackrel{\sigma}{\leftrightarrows} (F_1', p_1', q_1')$ and $R_2 : (F_2, p_2, q_2) \stackrel{\sigma}{\leftrightarrows} (F_2', p_2', q_2')$.

We will use the following extensions of the function successor: for $M \subseteq \mathbb{N}$, $S(M) = \{S(m) \mid m \in M\}$; and for $n \in \mathbb{N}$, $S^0(M) = M$ and $S^{n+1}(M) = S(S^n(M))$.

- $\sigma_{rel_{\mathcal{F}}}(F_1, p_1, q_1) \stackrel{\sigma}{\leftrightarrows} \sigma_{rel_{\mathcal{F}}}(F_1', p_1', q_1')$:
  take $R = \{(\{0\}, \{0\})\} \cup \{(S(M), S(M')) | (M, M') \in R_1\}$;
- $(F_1, p_1, q_1) \cdot_{\mathcal{F}} (F_2, p_2, q_2) \stackrel{\sigma}{\leftrightarrows} (F_1', p_1', q_1') \cdot_{\mathcal{F}} (F_2', p_2', q_2')$:
  take $R = \{(M, M') | (M, M') \in R_1\} \cup \{(S^{n_1}(M), S^{n_1'}(M') | (M, M') \in R_2\}$, where $n_1 = S(max(|F_1|))$ and $n_1' = (max(|F_1'|))$;
- $(F_1, p_1, q_1) +_{\mathcal{F}} (F_2, p_2, q_2) \stackrel{\sigma}{\leftrightarrows} (F_1', p_1', q_1') +_{\mathcal{F}} (F_2', p_2', q_2')$:
  take R to be the smallest relation satisfying the following conditions:
  - $(\{0\}, \{0\}) \in R$,
  - if $(M, M') \in R_1$, $(N, N') \in R_2$ and for some $k \in \mathbb{N}$, $M = \sigma^k(\{p_1\})$, $M' = \sigma^k(\{p_1'\})$, $N = \sigma^k(\{p_2\})$ and $N' = \sigma^k(\{p_2'\})$ then $(S(M) \cup S^{n_1}(N), S(M') \cup S^{n_1'}(N')) \in R$,
  - if $(M, M') \in R_1$ then $(S(M), S(M')) \in R$,
  - if $(N, N') \in R_2$ then $(S^{n_1}(M), S^{n_1'}(M')) \in R$,

  where $n_1 = S(S(max(|F_1|))$ and $n_1' = S(S(max(|F_1|)))$.

□

We define the following operation on graphs.

Definition 5.12 (frame-like sequential composition on graphs)
Given G_1 and G_2 graphs in $\mathcal{G}$ the graph $G_1 \cdot_{\mathcal{FG}} G_2$ is obtained by identifying all successful termination nodes in G_1. To the node obtained this way append the graph G_2 in which all successful termination nodes are identified. □

Lemma 5.13 $G_1 \cdot_{\mathcal{G}} G_2 \leftrightarrows G_1 \cdot_{\mathcal{FG}} G_2$.

Proof: Obvious. □

Lemma 5.14 $\mathsf{gr2fr}_{g_1}(G_1 \cdot_{\mathcal{FG}} G_2) \stackrel{\sigma}{\leftrightarrows} \mathsf{gr2fr}_{g_2}(G_1) \cdot_{\mathcal{F}} \mathsf{gr2fr}_{g_3}(G_2)$.

Proof: $\mathsf{ugr}(\mathsf{gr2fr}_{g_1}(G_1 \cdot_{\mathcal{FG}} G_2))$ is obtained by appending to each unsuccessful termination node of $G_1 \cdot_{\mathcal{FG}} G_2$ an outgoing σ-edge to a new node and identifying all termination nodes.

$\mathsf{ugr}(\mathsf{gr2fr}_{g_2}(G_1) \cdot_{\mathcal{F}} \mathsf{gr2fr}_{g_3}(G_2))$ is obtained as follows:

1. a σ-edge is appended to all unsuccessful termination nodes of G_1 and all termination nodes (including the new nodes which were added) are identified, thus obtaining a unique termination node;
2. the same operation is performed on G_2;
3. (a) the graph obtained in step (2) is appended to the node obtained in the identification performed in step (1);
   (b) the time steps which were appended to the unsuccessful termination nodes of G_1 and now have as target the termination node of G_1 are "re-directed" to the termination of G_2.

It is easy to see that $\mathsf{ugr}((\mathsf{gr2fr}_{g_1}(G_1 \cdot_{\mathcal{FG}} G_2))$ and $\mathsf{ugr}(\mathsf{gr2fr}_{g_1}(G_1) \cdot_{\mathcal{F}} \mathsf{gr2fr}_{g_2}(G_2))$ are isomorphic. A bisimulation between these graphs can be translated to a σ-bisimulation between $\mathsf{gr2fr}_{g_1}(G_1 \cdot_{\mathcal{FG}} G_2)$ and $\mathsf{gr2fr}_{g_1}(G_1) \cdot_{\mathcal{F}} \mathsf{gr2fr}_{g_2}(G_2)$. □

Lemma 5.15 $\mathsf{gr2fr}$ *is a homomorphism between* $(\mathcal{G}/\underline{\leftrightarrow}, +_{\mathcal{G}/\underline{\leftrightarrow}}, \cdot_{\mathcal{G}/\underline{\leftrightarrow}}, \sigma_{\mathrm{rel}_{\mathcal{G}/\underline{\leftrightarrow}}}, [\mathsf{cts}(a)_{\mathcal{G}}]_{\underline{\leftrightarrow}}, [\mathsf{cts}(\delta)_{\mathcal{G}}]_{\underline{\leftrightarrow}})$ *and* $(\mathcal{F}/\underline{\overset{\sigma}{\leftrightarrow}}, +_{\mathcal{F}/\underline{\overset{\sigma}{\leftrightarrow}}}, \cdot_{\mathcal{F}/\underline{\overset{\sigma}{\leftrightarrow}}}, \sigma_{\mathrm{rel}_{\mathrm{F}/\underline{\overset{\sigma}{\leftrightarrow}}}}, [\mathsf{cts}(a)_{\mathcal{F}}]_{\underline{\overset{\sigma}{\leftrightarrow}}}, [\mathsf{cts}(\delta)_{\mathcal{F}}]_{\underline{\overset{\sigma}{\leftrightarrow}}})$.

Proof: We have to prove that $\mathsf{gr2fr}$ commutes with the constants and operations of $\mathcal{G}/\underline{\leftrightarrow}$, i.e. that the following equivalences hold: $\mathsf{cts}(a)_{\mathcal{F}} \overset{\sigma}{\leftrightarrow} \mathsf{gr2fr}_g(\mathsf{cts}(a)_{\mathcal{G}})$, $\delta_{\mathcal{F}} \overset{\sigma}{\leftrightarrow} \mathsf{gr2fr}_g(\delta_{\mathcal{G}})$, $\sigma_{\mathsf{rel}}(\mathsf{gr2fr}_{g_1}(G)) \overset{\sigma}{\leftrightarrow} \mathsf{gr2fr}_{g_2}(\sigma_{\mathsf{rel}}(G))$ and for any two process graphs G_1 and G_2 in $\mathcal{G}$ we have: $\mathsf{gr2fr}_{g_1}(G_1 \cdot_{\mathcal{G}} G_2) \overset{\sigma}{\leftrightarrow} \mathsf{gr2fr}_{g_2}(G_1) \cdot_{\mathcal{F}} \mathsf{gr2fr}_{g_3}(G_2)$ and $\mathsf{gr2fr}_{g_1}(G_1 +_{\mathcal{G}} G_2) \overset{\sigma}{\leftrightarrow} \mathsf{gr2fr}_{g_2}(G_1) +_{\mathcal{F}} \mathsf{gr2fr}_{g_3}(G_2)$.

- (constants and σ_{rel})
  For the constants and the σ_{rel} operator the conclusions follow immediately from the definitions of constants and σ_{rel} and the definition of $\mathsf{gr2fr}$.
- (sequential composition)
  From Lemma 5.13 we have that $G_1 \cdot_{\mathcal{G}} G_2 \leftrightarrow G_1 \cdot_{\mathcal{FG}} G_2$. This means that $\mathsf{gr2fr}_{g_1}(G_1 \cdot_{\mathcal{G}} G_2) \overset{\sigma}{\leftrightarrow} \mathsf{gr2fr}_{g_1}(G_1 \cdot_{\mathcal{FG}} G_2)$ (Lemma 5.7). From this result combined with Lemma 5.14 we get $\mathsf{gr2fr}_{g_1}(G_1 \cdot_{\mathcal{G}} G_2) \overset{\sigma}{\leftrightarrow} \mathsf{gr2fr}_{g_2}(G_1) \cdot_{\mathcal{F}} \mathsf{gr2fr}_{g_3}(G_2)$.
- (alternative composition)
  Let G_1 and G_2 be two arbitrary graphs in $\mathcal{G}$. We distinguish four cases:

- $G_1 = G_2 = \mathsf{cts}(\delta)_{\mathcal{G}}$.
  It is a matter of simple calculations.
- $G_1 = \mathsf{cts}(\delta)_{\mathcal{G}}$ and $G_2 \neq \mathsf{cts}(\delta)_{\mathcal{G}}$.
  By definition, $\mathsf{cts}(\delta)_{\mathcal{G}} +_{\mathcal{G}} G_2 = G_2$. It follows that it is enough to prove that $\mathsf{gr2fr}_g(G_2) \overset{\sigma}{\leftrightarrow} \mathsf{gr2fr}_{g_1}(\mathsf{cts}(\delta)_{\mathcal{G}}) +_{\mathcal{G}} \mathsf{gr2fr}_{g_2}(G_2)$. For this, observe that $\mathsf{gr2fr}_{g_1}(\mathsf{cts}(\delta)_{\mathcal{G}}) = ((0 \xrightarrow{\sigma} g_1(u)), 0, g_1(u))$, where u is the only node in $\mathsf{cts}(\delta)_{\mathcal{G}}$. The σ-bisimulation between $\mathsf{gr2fr}_g(G_2)$ and $\mathsf{gr2fr}_{g_1}(\mathsf{cts}(\delta)_{\mathcal{G}}) +_{\mathcal{G}} \mathsf{gr2fr}_{g_2}(G_2)$ is R constructed as follows:
  $R = \{(g(r(G_2)), \{0\}\} \cup$
  $\quad \{(\{g(u)\}, \{S^{n_1}(g_2(u))\}) | u \in N(G_2) \setminus {\downarrow}(G_2)\}$,
  where $n_1 = S(g_1(u))$.
- $G_2 = \mathsf{cts}(\delta)_{\mathcal{G}}$ and $G_1 \neq \mathsf{cts}(\delta)_{\mathcal{G}}$.
  Similar to the above case.
- $G_1 \neq \mathsf{cts}(\delta)_{\mathcal{G}}$ and $G_2 \neq \mathsf{cts}(\delta)_{\mathcal{G}}$.
  We have to prove that $\mathsf{gr2fr}_{g_1}(G_1 +_{\mathcal{G}} G_2) \overset{\sigma}{\leftrightarrow} \mathsf{gr2fr}_{g_2}(G_1) +_{\mathcal{F}} \mathsf{gr2fr}_{g_3}(G_2)$ for two arbitrary graphs G_1 and G_2. For a graph G and $s, s' \in N(G)$ we will write $s \xrightarrow{\sigma^k} s'$ to mean that there is a path of length k starting from the node s, ending with the node s' and which contains only σ-edges. Take the following relation:
  $R = \{(\{g(\mathsf{idn}(r_1, r_2))\}, \{0\})\} \cup$
  $\quad \{(\{g(\mathsf{idn}(u, u'))\}, \{S(g_1(u)), S^{n_1}(g_2(u'))\}) | \exists k \in \mathbb{N}, r_1 \xrightarrow{\sigma^k} u, r_2 \xrightarrow{\sigma^k} u'\} \cup$
  $\quad \{(\{g(u)\}, \{S(g_1(u))\}) | u \in N(G_1) \setminus {\downarrow}(G_1)\} \cup$
  $\quad \{(\{g(u')\}, \{S^{n_1}(g_2(u'))\}) | u' \in N(G_2) \setminus {\downarrow}(G_2)\}$,

where $n_1 = S(\max\{g_1(u) \mid u \in N(G_1) \setminus {\downarrow}(G_1)\})$.

□

Theorem 5.16 *The graph model* $(\mathcal{G}/\underline{\leftrightarrow}, +_{\mathcal{G}/\underline{\leftrightarrow}}, \cdot_{\mathcal{G}/\underline{\leftrightarrow}}, \sigma_{\mathrm{rel}_{\mathcal{G}}/\underline{\leftrightarrow}}[\mathsf{cts}(a)]\underline{\leftrightarrow}, [\mathsf{cts}(\delta)]\underline{\leftrightarrow})$ *is isomorphic to the frame model* $(\mathcal{F}/\overset{\sigma}{\underline{\leftrightarrow}}, +_{\mathcal{F}/\overset{\sigma}{\underline{\leftrightarrow}}}, \cdot_{\mathcal{F}/\overset{\sigma}{\underline{\leftrightarrow}}}, \sigma_{\mathrm{rel}_{\mathcal{F}}/\overset{\sigma}{\underline{\leftrightarrow}}}[\mathsf{cts}(a)]_{\overset{\sigma}{\underline{\leftrightarrow}}}, [\mathsf{cts}(\delta)]_{\overset{\sigma}{\underline{\leftrightarrow}}})$.

Proof: The function $\mathsf{gr2fr}$ is a bijection on equivalence classes (Lemmas 5.9 and 5.10), and it is a homomorphism with respect to the operations (Lemma 5.15). □

6 Recursion in graphs and frames

In this section we give an interpretation of the constants $\langle X|E\rangle$, for finite linear recursive specifications E, on frames. First we extend the graph model of $\mathrm{BPA}^{=}_{\mathrm{drt}}$ for these constants. The resulting model satisfies RDP and RSP. After that we extend the frame model for these constants as well, resulting in a model isomorphic with the extended graph model.

6.1 Graph model for $\mathrm{BPA}^{=}_{\mathrm{drt}}\mathrm{Lin}$

The construction of the process graph corresponding to $\langle X|E\rangle$ in case of finite linear recursion follows the one for the untimed case (see for example [6]).

Let $X \in V$ and $E = E(V)$ be a finite linear recursive specification. Then the process graph $\langle X|E\rangle_{\mathcal{G}}$ is constructed as follows:

- there is a node in the graph for each variable $Y \in V$ (also be denoted by Y);
- the root of the graph is the node X;
- for each equation $Z = s_Z$ in $E(V)$:
  - for each summand $\mathsf{cts}(a_i) \cdot X_i$ in s_Z there is an edge labelled with a_i from the node Z to the node X_i,
  - for each summand $\mathsf{cts}(b_i)$ in s_Z there is an edge labelled with b_i from the node Z to a new node that is marked as a successful termination node,
  - for each summand $\sigma_{\mathsf{rel}}(Y_i)$ in s_Z there is an edge labelled with σ from the node Z to the node Y_i.

Theorem 6.1 *The graph model* $(G/\underline{\leftrightarrows}, +_{\mathcal{G}/\underline{\leftrightarrows}}, \cdot_{\mathcal{G}/\underline{\leftrightarrows}}, \sigma_{\mathrm{rel}_{\mathcal{G}/\underline{\leftrightarrows}}}, [\mathsf{cts}(a)_{\mathcal{G}}]_{\underline{\leftrightarrows}}, [\mathsf{cts}(\delta)_{\mathcal{G}}]_{\underline{\leftrightarrows}}, [\langle X|E\rangle_{\mathcal{G}}]_{\underline{\leftrightarrows}})$ *satisfies* $\mathrm{BPA}^{=}_{\mathrm{drt}}\mathrm{Lin}$+RDP.

Proof: We show that, if $E = E(V)$ is a recursive specification and $X = \sum_i \mathsf{cts}(a_i) \cdot X_i + \sum_j \mathsf{cts}(b_i) + \sigma_{\mathsf{rel}}(Y)$ is an equation of E, then

$$\langle X|E\rangle_{\mathcal{G}} \leftrightarrows \textstyle\sum_i \mathsf{cts}(a_i)_{\mathcal{G}} \cdot \langle X_i|E\rangle_{\mathcal{G}} + \sum_j \mathsf{cts}(b_j)_{\mathcal{G}} + \sigma_{\mathsf{rel}}(\langle Y|E\rangle_{\mathcal{G}})$$

holds. *RDP* follows then immediately.

The right hand side of the equation is the alternative composition of several graphs. First there are the graphs of the form $\mathsf{cts}(a_i)_{\mathcal{G}}\langle X_i|E\rangle_{\mathcal{G}}$. Then there are the graphs $\mathsf{cts}(b_i)_{\mathcal{G}}$ and finally there is the graph $\sigma_{\mathsf{rel}}(\langle Y|E\rangle_{\mathcal{G}})$. Among these graphs only the latest has an outgoing σ-edge from its root. This implies that step (b) from the definition of alternative composition on graphs will not be applied. Due to this fact calculating the alternative composition of these graph is just unwinding their roots and identifying all the new roots. A general picture is presented in Figure 2. The picture also gives a clue on how a bisimulation between the left hand side graph and the right hand side graph can be obtained, namely a relation which relates the node X in the graph $\langle X|E\rangle_{\mathcal{G}}$ with all its copies and with the root r of the graph on the right hand side, each node $Z \in V \setminus \{X\}$ with all its copies on the right hand side, and a successful termination node on the left hand side with all its copies on the right hand side. □

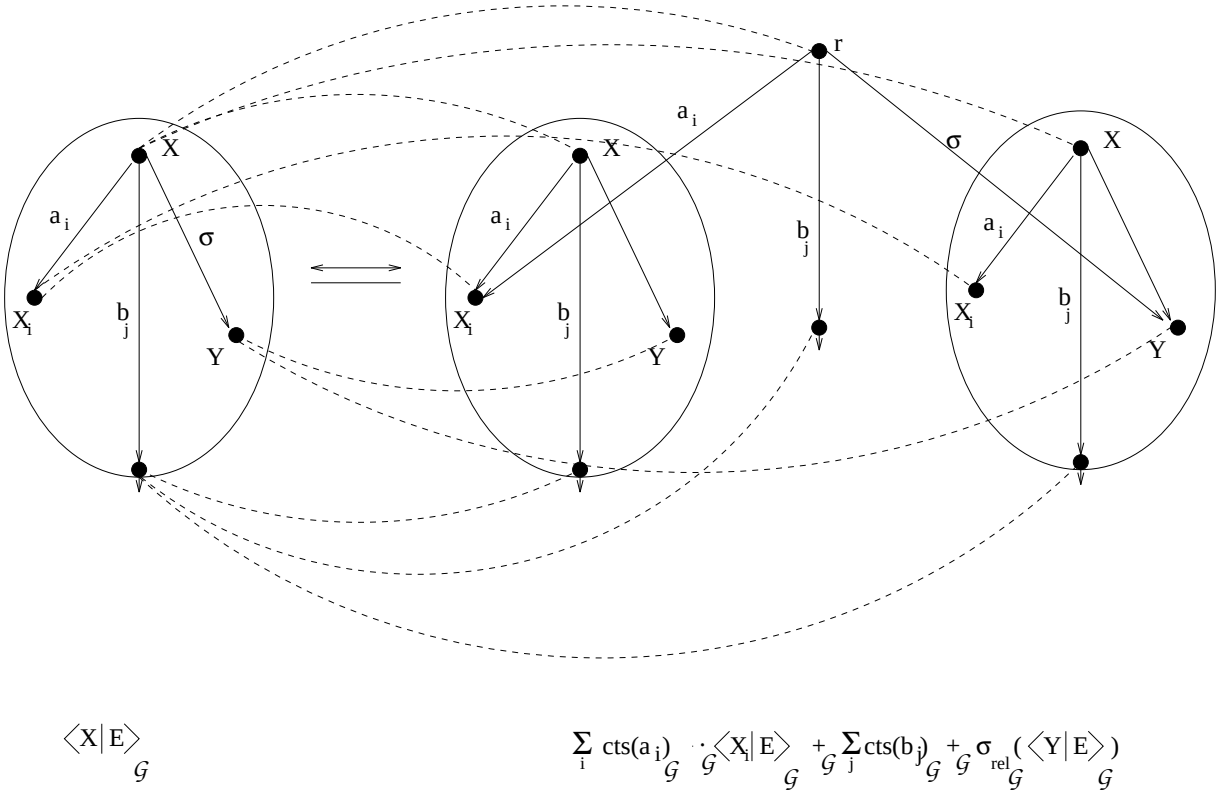

Figure 2: The graph model satisfies Lin

We will prove now that the graph model for discrete time process algebra with recursion satisfy RSP.

For this we introduce a projection operator for discrete time process algebra without immediate deadlock. The axioms which define this operator are given in Table 6.

$\pi_0(X) = \mathsf{cts}(\delta)$	PR1	$\pi_{n+1}(\mathsf{cts}(a)) = \mathsf{cts}(a)$	PR4
$\pi_n(\mathsf{cts}(\delta)) = \mathsf{cts}(\delta)$	PR2	$\pi_{n+1}(\mathsf{cts}(a) \cdot X) = \mathsf{cts}(a) \cdot \pi_n(X)$	PR5
$\pi_{n+1}(\sigma_{\mathsf{rel}}(X)) = \sigma_{\mathsf{rel}}(\pi_n(X))$	PR3	$\pi_n(X + Y) = \pi_n(X) + \pi_n(Y)$	PR6

Table 6: Axioms of the projection operator

We can formulate now the AIP (Approximation Induction Principle). Informally it states that the processes are uniquely defined by their finite projections.

Definition 6.2 (AIP)
A model is said to satisfy AIP if for any x and y the following statement is true:

$$(\forall n \geq 1, \pi_n(x) = \pi_n(y)) \Rightarrow x = y.$$

□

We also define the head normal form of a process.

Definition 6.3 (head normal form)
We say a process expression p has a head normal form if $p = \mathsf{cts}(\delta)$ or there are $k, l, m \in \mathbb{N}$, with $k + l + m > 0$, atomic actions a_i $(i \leq k)$, b_i $(i \leq l)$ and processes p_i $(i \leq k)$, q_i $(i \leq m)$ such that $p = \sum_{i<k} cts(a_i) \cdot p_i + \sum_{i<l} cts(b_i) + \sum_{i<m} \sigma_{\mathsf{rel}}(q_i)$, □

Using this notions we will prove that a model satisfying AIP also satisfies RSP.

Lemma 6.4 *Every process expression in $BPA^{=}_{\mathrm{drt}}Lin$ has a head normal form.*

Proof: We do the proof by structural induction on the terms of $\mathrm{BPA}^{=}_{\mathrm{drt}}\mathrm{Lin}$. For the constants it is obvious. Consider that the processes p and p' have as head normal form:

- $p = \sum_{i<k_1} \mathsf{cts}(a_i) \cdot p_i + \sum_{i<l_1} \mathsf{cts}(b_i) + \sum_{i<m_1} \sigma_{\mathsf{rel}}(q_i)$
- $p' = \sum_{i<k_2} \mathsf{cts}(a'_i) \cdot p'_i + \sum_{i<l_2} \mathsf{cts}(b'_i) + \sum_{i<m_2} \sigma_{\mathsf{rel}}(q'_i)$

The head normal form of processes are:

- (σ_{rel})
  $\sigma_{\mathsf{rel}}(p)$ is a head normal form already;
- (alternative composition)
  $p + p' = \sum_{i<k_1} \mathsf{cts}(a_i) \cdot p_i + \sum_{i<k_2} \mathsf{cts}(a'_i) \cdot p'_i + \sum_{i<l_1} \mathsf{cts}(b_i) + \sum_{i<l_2} \mathsf{cts}(b'_i) + \sum_{i<m_1} \sigma_{\mathsf{rel}}(q_i) + \sum_{i<m_2} \sigma_{\mathsf{rel}}(q'_i)$;
- (sequential composition)
  $p \cdot p' = \sum_{i<k_1} \mathsf{cts}(a_i) \cdot p_i \cdot p' + \sum_{i<l_1} \mathsf{cts}(b_i) \cdot p' + \sum_{i<m_1} \sigma_{\mathsf{rel}}(q_i \cdot p')$.

We conclude that all processes in $\mathrm{BPA}^{=}_{\mathrm{drt}}\mathrm{Lin}$ have a head normal form. Moreover, every p in $\mathrm{BPA}^{=}_{\mathrm{drt}}\mathrm{Lin}$ has a head normal form $\sum_{i<k} \mathsf{cts}(a_i) \cdot p_i + \sum_{i<l} \mathsf{cts}(b_i) + \sum_{i<m} \sigma_{\mathsf{rel}}(q_i)$ with all the p_i and p_j processes in $\mathrm{BPA}^{=}_{\mathrm{drt}}\mathrm{Lin}$. □

Theorem 6.5 *All projections of a process in $BPA^{=}_{\mathrm{drt}}Lin$ are equal to a closed term.*

Proof: By induction on k it is proved that $\pi_k(p)$ is equal to a closed term. The proof is as in [6] with small, obvious modifications. □

Corollary 6.6 *If E is a linear recursive specification with solutions p and q then for all $n > 0$ we have $\pi_n(p) = \pi_n(q)$*

Theorem 6.7 *A model which satisfies AIP satisfies RSP.*

Proof: Suppose that a linear recursive specification has solutions p and q. By 6.5 we have that $\pi_n(p) = \pi_n(q)$ for all $n \geq 0$. Then it follows from AIP that $p = q$. □

We prove that the graph model for discrete time process algebra with recursion satisfies AIP.

Definition 6.8 ($\mathsf{tree}(G)$)
For a graph $G \in \mathcal{G}$, $\mathsf{tree}(G)$ is defined as follows:

1. tree(G) has a node for every finite path of G, that starts at the root of G. Such a node has label $\downarrow$ if the last node of the path does.
2. There is an edge labelled with a or σ between p and p' if the path p' is an extension of the path p with an edge labelled a or σ respectively.
3. The root of $\mathsf{tree}(G)$ is the node corresponding to the empty path in G.

□

Remark 6.9 Observe that $\mathsf{tree}(G)$ is not necessarily a process graph. If we extend the definition of bisimulation to infinite graphs, then it is true that $G \leftrightarroweq \mathsf{tree}(G)$. □

Definition 6.10 (depth of a node in $\mathsf{tree}(G)$)
If G is a process graph then for each node p of tree (G) we define *depth* of p as being the length of the path to which it corresponds. □

Definition 6.11 ($\pi_n(G)$)
Let G be a process graph. We obtain $\pi_n(G)$ as follows:

1. Take $\mathsf{tree}(G)$
2. Remove all edges leaving from a node at depth n.

□

Lemma 6.12 $(\mathcal{G}/\underline{\leftrightarrows}, +_{\mathcal{G}/\underline{\leftrightarrows}}, \cdot_{\mathcal{G}/\underline{\leftrightarrows}}, \sigma_{\mathrm{rel}_{\mathrm{G}/\underline{\leftrightarrows}}}, [\mathsf{cts}(a)_{\mathcal{G}}]\underline{\leftrightarrows}, [\mathsf{cts}(\delta)_{\mathcal{G}}]\underline{\leftrightarrows}, [\langle X|E\rangle_{\mathcal{G}}]\underline{\leftrightarrows})$ *satisfies AIP.*

Proof: Consider G_1 and G_2 such that $\pi_n(G_1) \mathrel{\underline{\leftrightarrows}} \pi_n(G_2)$ for all $n \in \mathbb{N}$. We have to prove that $G_1 \mathrel{\underline{\leftrightarrows}} G_2$. We will prove that $\mathsf{tree}(G_1) \mathrel{\underline{\leftrightarrows}} \mathsf{tree}(G_2)$. Consider the set of all bisimulation relations between $\pi_n(G_1)$ and $\pi_n(G_2)$ for all n. We introduce a partial order relation on these bisimulations. If R is a bisimulation between $\pi_n(G_1)$ and $\pi_n(G_2)$ and R' a bisimulation between $\pi_{n+1}(G_1)$ and $\pi_{n+1}(G_2)$ then there is an edge between R and R' if $R \subseteq R'$. We obtain an finitely branching infinite tree having as labels bisimulation relations. Due to König's lemma there is an infinite path which starts from the root. Take the union of the bisimulation relation along this path. This will yield a bisimulation relation between $\mathsf{tree}(G_1)$ and $\mathsf{tree}(G_2)$. Using Remark 6.9 we obtain that $G_1 \mathrel{\underline{\leftrightarrows}} G_2$. □

Theorem 6.13 $(\mathcal{G}/\underline{\leftrightarrows}, +_{\mathcal{G}/\underline{\leftrightarrows}}, \cdot_{\mathcal{G}/\underline{\leftrightarrows}}, \sigma_{\mathrm{rel}_{\mathrm{G}/\underline{\leftrightarrows}}}, [\mathsf{cts}(a)_{\mathcal{G}}]\underline{\leftrightarrows}, [\mathsf{cts}(\delta)_{\mathcal{G}}]\underline{\leftrightarrows}, [\langle X|E\rangle_{\mathcal{G}}]\underline{\leftrightarrows})$ *satisfies RSP.*

Proof: It follows from 6.7 and previous lemma. □

6.2 Timed frame model for $\mathrm{BPA}_{\mathrm{drt}}^{=}$Lin

Next we give an interpretation in $\mathcal{F}$ to the constants $\langle X|E\rangle$ for finite linear recursive specifications E.

Let $X \in V$, let $E = \{X_1 = s_{X_1}, ..., X_n = s_{X_n}\}$ be a linear recursive specification and let $e : V \to \mathbb{N}_1$ be an enumeration of the variables V. Then the interpretation of the constant $\langle X|E\rangle$ with respect to e is given by the definition in Table 7. The specific enumeration used is

$$\mathcal{F}_e(\langle X|E\rangle) \quad = \quad (0 \oplus \bigoplus_{i=1}^{n} \mathcal{F}_e(X_i = E_{X_i}), e(X), 0)$$

where

$$\mathcal{F}_e(Z = \mathsf{cts}(\delta)) = (e(Z) \xrightarrow{\sigma} 0),$$

$$\mathcal{F}_e(Z = \textstyle\sum_{i=1}^{k} \mathsf{cts}(a_i) \cdot X_i + \sum_{i=1}^{l} \mathsf{cts}(b_i)) = \bigoplus_{i=1}^{l}(e(Z) \xrightarrow{a_i} e(X_i)) \oplus \bigoplus_{i=i}^{m}(e(Z) \xrightarrow{b_i} 0),$$

$$\mathcal{F}_e(Z = \textstyle\sum_{i=1}^{k} \mathsf{cts}(a_i) \cdot X_i + \sum_{i=1}^{l} \mathsf{cts}(b_i) + \sum_{i=1}^{m} \sigma_{\mathsf{rel}}(Y_i)) = \bigoplus_{i=1}^{k}(e(Z) \xrightarrow{a_i} e(X_i)) \oplus \bigoplus_{i=1}^{l}(e(Z) \xrightarrow{b_i} 0) \oplus \bigoplus_{i=1}^{m}(e(Z) \xrightarrow{\sigma} e(Y_i))$$

Table 7: Interpretation of the constants $\langle X|E\rangle$

not important.

Lemma 6.14 *If $E = E(V)$ is a finite linear recursive specification and e and e' are two enumerations of the variables in V then the frames $\mathcal{F}_e(\langle X|E\rangle)$ and $\mathcal{F}_{e'}(\langle X|E\rangle)$ are σ-bisimilar.*

Proof: Define $f(e(X)) = e'(X)$ and $f(0) = 0$. It is easy to see that f is an isomorphism between $\mathcal{F}_e(\langle X|E\rangle)$ and $\mathcal{F}_{e'}(\langle X|E\rangle)$. According to 4.6 $\mathcal{F}_e(\langle X|E\rangle) \stackrel{\sigma}{\leftrightarrow} \mathcal{F}_{e'}(\langle X|E\rangle)$. □

Lemma 6.15 *If $E = E(V)$ is a recursive specification and $X \in V$ and e and g are two arbitrary enumeration then* $\mathsf{gr2fr}_g(\langle X|E\rangle_{\mathcal{G}}) \stackrel{\sigma}{\leftrightarrow} \mathcal{F}_e(\langle X|E\rangle)$.

Proof: Let Y be the node attached to a variable $Y \in V$ in the graph $\langle X|E\rangle_{\mathcal{G}}$ and let q be the node obtained in step (b) of $\mathsf{gr2fr}_g$ when applied to $\langle X|E\rangle_{\mathcal{G}}$. Abusing notation we will denote by $\mathsf{gr2fr}_g(\langle X|E\rangle)$ the frame which corresponds to the pointed frame $\mathsf{gr2fr}_g(\langle X|E\rangle)$.

Define $f : \mathsf{gr2fr}_g(\langle X|E\rangle_{\mathcal{G}}) \to \mathcal{F}_e(\langle X|E\rangle)$ by $f(g(q)) = 0$ and $f(g(Y)) = e(Y)$. Obviously it is a bijection between the states of the two frames. Observe that:

- $[e(Y) \xrightarrow{a} e(Z)]_{\mathcal{F}_e(\langle X|E\rangle)}$ iff there is a summand of the form $\mathsf{cts}(a) \cdot Z$ in E_Y iff there is an edge $Y \xrightarrow{a} Z$ in $\langle X|E\rangle_G$ iff $[g(Y) \xrightarrow{a} g(Z)]_{\mathsf{gr2fr}_g(\langle X|E\rangle_{\mathcal{G}})}$
- $[e(Y) \xrightarrow{\sigma} e(Z)]_{\mathcal{F}_e(\langle X|E\rangle)}$, iff there is a summand of the form $\sigma_{\mathsf{rel}}(Z)$ in E_Y iff there is an edge $Y \xrightarrow{\sigma} Z$ in $\langle X|E\rangle_G$ iff $[g(Y) \xrightarrow{\sigma} g(Z)]_{\mathsf{gr2fr}_g(\langle X|E\rangle_{\mathcal{G}})}$.
- $[e(Y) \xrightarrow{b} 0]_{\mathcal{F}_e(\langle X|E\rangle)}$ iff there is a summand of the form $\mathsf{cts}(b)$ in E_Y iff in $\langle X|E\rangle_{\mathcal{G}}$ there is an edge labelled with b from the node Y to a successful termination node iff $[g(Y) \xrightarrow{b} g(q)]_{\mathsf{gr2fr}_g(\langle X|E\rangle_{\mathcal{G}})}$.
- $[e(Y) \xrightarrow{\sigma} 0]_{\mathcal{F}_e(\langle X|E\rangle)}$ iff E_Y is $\mathsf{cts}(\delta)$ iff in $\langle X|E\rangle_{\mathcal{G}}$ the node Y is an unsuccessful termination node iff $[g(Y) \xrightarrow{\sigma} g(q)]_{\mathsf{gr2fr}_g\langle X|E\rangle}$.

This indicates that f as defined above is an isomorphism between $\mathsf{gr2fr}_g(\langle X|E\rangle_{\mathcal{G}})$ and $\mathcal{F}_e(\langle X|E\rangle)$. □

We define $\langle X|E\rangle_{\mathcal{F}}$ as $\mathcal{F}_e(\langle X|E\rangle)$ for some enumeration e. According to 6.14, $\mathcal{F}_e(\langle X|E\rangle)$ depends on e only modulo σ-bisimulation.

The given interpretation of constants $\langle X|E\rangle$ on frames is an algebraic re-formulation of the construction of the corresponding process graphs.

Theorem 6.16
$(G/\underline{\leftrightarrow}, +_{\mathcal{G}/\underline{\leftrightarrow}}, \cdot_{\mathcal{G}/\underline{\leftrightarrow}}, \sigma_{\mathrm{rel}_{\mathcal{G}/\underline{\leftrightarrow}}}, [\mathsf{cts}(a)_{\mathcal{G}}]\underline{\leftrightarrow}, [\mathsf{cts}(\delta)_{\mathcal{G}}]\underline{\leftrightarrow}, [\langle X|E\rangle_{\mathcal{G}}]\underline{\leftrightarrow})$ *is isomorphic to*

$(\mathcal{F}/\stackrel{\sigma}{\leftrightarrow}, +_{\mathcal{F}/\underline{\stackrel{\sigma}{\leftrightarrow}}}, \cdot_{\mathcal{F}/\underline{\stackrel{\sigma}{\leftrightarrow}}}, \sigma_{\mathrm{rel}_{\mathrm{F}/\underline{\leftrightarrow}}}, [\mathsf{cts}(a)_{\mathcal{F}}]\underline{\stackrel{\sigma}{\leftrightarrow}}, [\mathsf{cts}(\delta)_{\mathcal{F}}]\underline{\stackrel{\sigma}{\leftrightarrow}}, [\langle X|E\rangle_{\mathcal{F}}]\underline{\stackrel{\sigma}{\leftrightarrow}})$

Proof: It follows from 5.16 and the previous lemma that $\mathsf{gr2fr}$ is an isomorphism between these models. □

7 Conclusions and future work

Using the simple algebraic setting for timed frames, we have built a model of $\mathrm{BPA}^{\equiv}_{\mathrm{drt}}$-IDlin. This model can be extended to include other features, such as propositions and conditions.

Acknowledgement

The third author would like to thank to Radu Şoricuţ and Yaroslav Usenko for their usefull comments which led to this form of the paper.

References

[1] G.M. Reed ad A.W. Roscoe. A timed model for communication sequential processes. *TCS*, (58):249–261, 1988.

[2] J.C.M. Baeten and J.A. Bergstra. Real time process algebra. *Formal Aspects of Computing*, 3(2):142–188, 1991.

[3] J.C.M. Baeten and J.A. Bergstra. Discrete time process algebra. *Formal Aspects of Computing*, 8:188–208, 1996.

[4] J.C.M. Baeten and J.A.Bergstra. Discrete Time Process Algebra:absolute time, relative time and parametric time. *Fundamenta Informaticae*, 29:51–76, 1997.

[5] J.C.M. Baeten and M.A. Reniers. Discrete Time Process Algebra with Relative Timing. draft.

[6] J.C.M. Baeten and W.P. Weijland. *Process Algebra.* Cambridge Tracts in Theoretical Computer Science 18, Cambridge University Press, 1990.

[7] J.A. Bergstra, W.J. Fokkink, and C.A. Middelburg. Algebra of timed frames. *International Journal of Computer Mathematics*, 61:227–255, 1996.

[8] J.A. Bergstra and A. Ponse. Frame algebra with synchronous communication. In R.J. Wieringa and R.B. Feenstra, editors, *Information Systems – Correctness and Reusability*, pages 3–15. World Scientific, 1995.

[9] F. Moller and C. Tofts. A temporal calculus of communicating systems. In *CONCUR'90*, LNCS. Springer, 1990.