Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

J.A. Bergstra, J.W. Klop, E.-R. Olderog

Failure semantics with fair abstraction

Computer Science/Department of Software Technology Report CS-R8609 January

The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

Failure Semantics with Fair Abstraction

J.A. Bergstra
Universiteit van Amsterdam; Rijksuniversiteit Utrecht

J.W. Klop
Centre for Mathematics and Computer Science

E.-R. Olderog
Christian-Albrechts Universität, Kiel

1980 Mathematics Subject Classification: 68B10, 68C01, 68D25, 68F20.
1982 CR Categories: F.1.1, F.1.2, F.3.2, F.4.3. 69F11, 69F12, 69F32, 69F43.
Key Words & Phrases: process algebra, concurrency, failure semantics, bisimulation semantics.
Note: The research of J.A. Bergstra and J.W. Klop is partially supported by ESPRIT project 432: Meteor.
This report will be submitted for publication elsewhere.

We consider countably branching processes subject to operations: alternative composition (+), sequential composition (•) and abstraction (τ_I). Parallel operators are not yet considered. Axiom systems are given for such processes which contain δ (deadlock), ϵ (empty process), τ (silent move) and Δ (delay). The emphasis is on axiomatising divergence via the new constant Δ, both in the context of bisimulation semantics and failure semantics. A new process model is found which is intermediate between bisimulation semantics with fair abstraction and failure semantics with catastrophic divergence.

Report CS-R8609
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

INTRODUCTION

This paper is a sequel to BERGSTRA, KLOP & OLDEROG [8] where a study was made of failure semantics in the context of ACP, Algebra of Communicating Processes. To modularize the problems, two restrictions were adopted in [8]: only finite processes were considered (no recursion) and τ-steps were left aside (no abstraction). It turned out to be worth-while to study failure semantics even in this simple setting, and it was proved in [8] that failure equivalence is the maximal 'trace-respecting' congruence on finite process terms.

The present work removes both restrictions and is concerned with failure semantics for infinite processes with τ-steps. (However, we adopt another restriction in that we do not yet admit parallel operators.) Here we profit greatly from the work of BROOKES [9], who gave a complete axiomatisation of failure equivalence on finite CCS-expressions. Brookes' paper establishes a useful connection between the notion of failure semantics, arising from the work on CSP (see BROOKES, HOARE & ROSCOE [10]), and the synchronisation trees of Milner's CCS (see [17]). As Brookes shows, the notion of failure semantics applies perfectly well to an underlying domain of synchronisation trees - or process graphs as we prefer to call them in the setting of ACP. Moreover, BROOKES [9] presents some elegant axioms describing the failure equivalence, comparable to the well-known and beautiful τ-laws of Milner for bisimulation semantics. In fact, the bisimulation τ-laws are implied by the failure axioms, as stated in [9]. Brookes' work is therefore in our view one of the first attempts to give a uniform algebraic view on a spectrum of theories about communicating processes, ranging from CCS-like theories based upon observational equivalence or bisimulation, to CSP-like theories based upon notions such as readiness or failure semantics.

Striving towards a uniform view on algebraic theories of communicating processes is also an endeavour of the present work, where we always will work in the framework of ACP_τ, Algebra of Communicating Processes with abstraction, as introduced in BERGSTRA & KLOP [7]. Actually we use only a fragment of ACP_τ, namely BPA_δ + T1-3 + TI1-5; this is the axiom system in Table 1 after deleting TI6 and the rule DE, consisting of axioms for + and $\cdot$ (sum, as in CCS, resp. product or sequential composition), axioms for deadlock (δ), Milner's τ-laws T1-3 and axioms for abstraction (τ_I). Introducing the parallel operator $\|$ of does not seem to present a real problem but is postponed for the time being in order to concentrate on even more basic issues.

One of the most important issues in a study of semantics for infinite processes with τ-steps (bisimulation, failure or other semantics) is the way in which *divergence* is treated. Here 'divergence' is taken to be the capability of a process of executing an infinite sequence of τ-steps. In fact, a major motivation for the present work was the treatment in BROOKES, HOARE & ROSCOE [10] of divergence as being catastrophic: there all processes with an infinite τ-trace from the root are identified (with the wholly arbitrary process CHAOS). Somehow this is a disappointing solution, and it is intriguing to see whether the very notion of failure semantics forces one to adopt this solution; [10] contains some arguments which seem to suggest this. That this solution is disappointing is because it greatly reduces the attractivity of doing system verification using failure semantics (with CHAOS): cf. BERGSTRA & KLOP [5] where a simple Alternating Bit Protocol was verified on the basis of Koomen's Fair Abstraction Rule (KFAR) which allows one to abstract from divergence in a 'graceful' way rather than seeing the whole process becoming absorbed by the black hole CHAOS after which nothing is left to verify.

A related problem concerning divergence is found in 'classical' CCS (i.e. as in MILNER [18]). There it turns out (p.99) that the infinite sequence of τ-steps, τ^ω, is identified by observational equivalence with the zero process NIL. As Milner states, it is possible to adapt the notion of observational equivalence in such a way that the presence of infinite unseen action is respected. Also BROOKES [9] states in his conclusions that the "inability to distinguish between divergence and deadlock is unappealing".

The present notes give axiomatisations which pertain to just those questions, namely how divergence can be treated. There turns out to be an interesting bifurcation, or rather trifurcation, in the development of theories about communicating processes. There are three main theories (axiomatisations) which are mutually incompatible:

- bisimulation semantics for infinite processes with τ-steps satisfying KFAR; here $\tau^\omega = \tau\delta$ (in CCS we would have τ^ω = NIL),
- failure semantics for infinite processes with τ-steps allowing abstraction from 'unstable divergence', to be explained,

- failure semantics for infinite processes with τ-steps satisfying $\tau^{\omega} =$ CHAOS (as in BROOKES, HOARE & ROSCOE [10]).

The second theory is the main topic of this paper. The incompatibility of the first and the third theory was demonstrated in a closing section in BERGSTRA, KLOP & OLDEROG [8]: there it was shown that KFAR, combined with even a small fragment of the failure semantics axiomatisation, yields an inconsistency. (Here 'consistent' means 'trace-respecting'; a precise definition is given in Section 1.4.)

Both the exposition and the elegance of the various axiomatisations are greatly enhanced by introducing an operator Δ, for 'delay', which in the notation of μ-calculus can be expressed by $\Delta x = \mu Y(\tau Y + x)$. So Δx is "x with a τ-loop appended at the root". Of course, the delay operator is not new here: it plays a key role in MILNER [19] where it is written as δ and is used to simulate asynchronous processes by means of synchronous ones. The difference is that our Δ refers to τ while the delay operator in Milner's work refers to $\mathbb{1}$ in SCCS, which has properties quite different from τ. With the delay operator, the typical and mutually irreconcilable differences between the three main theories indicated above can now simply be stated:

- $\Delta = \tau$
- $\Delta\tau = \tau$
- $\Delta\delta = \Delta$.

So Δ acts like a discriminant between these theories. The expression $\Delta\tau$ stands for 'unstable divergence': divergence with the possibility of a silent exit from it.

As said, the second possibility receives most attention in these notes. (The first one, bisimulation semantics for infinite processes with τ-steps together with the rule KFAR, is extensively explored in BAETEN, BERGSTRA & KLOP [1], albeit without the explicit notation Δ for divergence, which is not necessary there since KFAR says in effect that $\Delta = \tau$.) In fact, we provide a model for this axiom system which has a domain of countably branching process graphs as basic building material. Process graphs are transition diagrams where the edges are labeled with symbols from some alphabet of atomic actions (events, steps). After unwinding, the resulting process trees are often called 'synchronisation trees', by various authors starting with Milner. Our definition of the failure semantics of a process graph parallels the usual one, in BROOKES, HOARE & ROSCOE [10], for finite processes, and thereby the one in BERGSTRA, KLOP & OLDEROG [8] (but for a minor detail necessary for the extension to τ-steps); moreover, account is taken whether the process graph is 'stable' or 'unstable'. This terminology is from MILNER [18], where a process having a τ-step as an initial step is called unstable. In failure semantics it is easy to derive that a process x is unstable iff $x = \tau x$; and it turns out that the processes considered in BROOKES, HOARE & ROSCOE [10], when formulated with CCS primitives + and τ (as done in BROOKES [9]) can be taken as the set of unstable processes. Taking note of the (un)stability of a process in its failure semantics removes one slightly irritating phenomenon, encountered in MILNER [18] and in BROOKES [9], namely that the equivalence would otherwise not be a congruence (the well-known point being that $\tau x \approx x$ but not $\tau x + y \approx x + y$). This may seem a minor point but it is important for a proper algebraical treatment of these matters to work only with congruences.

The model that we provide for failure semantics with abstraction from unstable divergence ($\Delta\tau = \tau$), is based on a very natural definition of failure equivalence. Two countably branching process graphs g,h are failure equivalent if their failure sets coincide; the failure set of g contains next to the prefix closure of all finite traces of g, only failure pairs obtained from stable nodes of g (a node is stable if no τ-step leaves from it). Remarkably, the definition does not mention divergence at all. (Definition 4.2.2.)

We briefly consider process algebras (in bisimulation semantics with fair abstraction as well as failure semantics with fair abstraction of unstable divergence) consisting only of 'characteristic' processes, i.e. processes built from δ, ε, τ, Δ. An insight into the structure of such a process algebra is an insight in the "termination characteristics" of the semantics in question, since all the constants δ, ε, τ, Δ have something to do with termination or non-termination. Here a surprising fact was encountered by KOYMANS & VRANCKEN [7]: in bisimulation semantics with fair abstraction ($\Delta = \tau$), this algebra of δ, ε, τ-processes has the structure of the so-called Rieger-Nishimura lattice in intuitionistic propositional logic.

Acknowledgement. We thank R. van Glabbeek for detecting a serious shortcoming in an earlier version of this paper, and for his suggestions for remedies.

CONTENTS

1. BISIMULATION SEMANTICS WITH EXPLICIT DIVERGENCE: BS_{Δ}

1.1. *Axioms.*

The first axiom system is introduced as it is a kernel system for all later axiom systems in this paper. The axioms and rules are included in all subsequent systems - though some axioms will disappear in later systems as they become derivable there. The axioms and rules for *bisimulation semantics with explicit divergence*, BS_{Δ} for short, are as in Table 1.

TABLE 1. BS_{Δ}, bisimulation semantics with explicit divergence

$x + y = y + x$	A1
$(x + y) + z = x + (y + z)$	A2
$x + x = x$	A3
$(x + y)z = xz + yz$	A4
$(xy)z = x(yz)$	A5
$\delta + x = x$	A6
$\delta x = \delta$	A7
$a\tau = a$	T1
$\tau\tau = \tau$	T1'
$\tau x + x = \tau x$	T2
$a(\tau x + y) = a(\tau x + y) + ax$	T3
$\tau_I(\tau) = \tau$	TI1
$\tau_I(a) = a \quad \text{if } a \notin I$	TI2
$\tau_I(a) = \tau \quad \text{if } a \in I$	TI3
$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$	TI4
$\tau_I(xy) = \tau_I(x).\tau_I(y)$	TI5
$\tau_I(\Delta(x)) = \Delta(\tau_I(x))$	TI6
$\dfrac{\forall k \in \mathbb{Z}_n \quad x_k = i_k x_{k+1} + y_k \quad (i_k \in I)}{\tau_I(x_0) = \Delta(\tau_I(\sum_{k \in \mathbb{Z}_n} y_k))}$	$DE_n \ (n \geqslant 1)$

Here A1-7 is BPA_δ, T1-3 are Milner's τ-laws and TI1-6 describe the working of the abstraction operators τ_I where $I \subseteq A$; A is the set of atoms (steps, actions). We write $A_\delta = A \dot{\cup} \{\delta\}$. The axiom system so far has been introduced and discussed in [1,3,4,8].

New is the unary operator Δ, called *delay*, subject to the *delay rule* DE. The DE rule is parametrised by $k \geqslant 1$, and for $k=1$ the rule 'introduces' Δ as follows:

$$DE_1 \qquad \frac{x = ix + y, \quad i \notin \alpha(y)}{\tau_{\{i\}}(x) = \Delta(y)}$$

That is, $\Delta(y)$ is the result of appending a "τ-loop at the root of y". The mechanics of appending this τ-loop are as follows: for a fresh i ($i \notin \alpha(y)$, the alphabet used by y) determine x with $x = ix + y$, and abstract from i (i.e. rename i into τ). Since x is uniquely determined by $x = ix + y$, the result $\tau_{\{i\}}(x)$ is well-defined. Note that the detour via i and $\tau_{\{i\}}$ is necessary since the recursion equation $x = \tau x + y$ would not define x uniquely as every $\tau(y + q)$, q arbitrary, is a solution.

1.1.1. REMARK. Another device to overcome the problem of τ-guarded recursion equations with their non-unique solutions, is adopted by MILNER [20] and consists of the use of μ-expressions where so to say our abstraction operator τ_I is already 'built in'. Then one defines

$$\Delta(y) = \mu X[\tau X + y].$$

Indeed a delay operator is introduced in this way in MILNER [19], though in a different setting, without τ-laws.

For $k \geqslant 1$, the rules DE_k perform a contraction of a τ-cycle of length k into a τ-loop. (We use 'loop' for 'cycle of length 1'.) For example, if

$$x = iy + a, \quad y = jz + b, \quad z = kx + c$$

then DE_3 allows the conclusion

$$\tau_{\{i,j,k\}}(x) = \Delta(a + b + c).$$

In a picture:

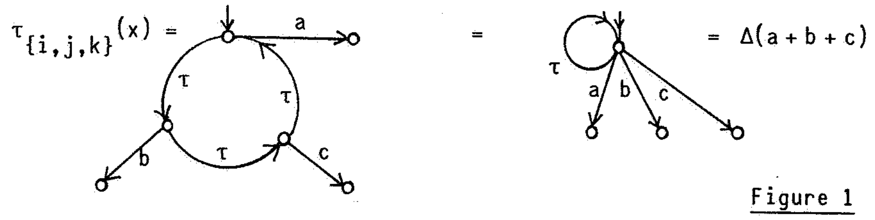


Figure 1

Figure 1 anticipates the description in 1.2 below of the model $\mathcal{A}(BS_\Delta)$ for the axiom system in Table 1. First let us note some consequences of the axioms and rules:

1.1.2. PROPOSITION.

(i) $\Delta(x)\cdot y = \Delta(x\cdot y)$

(ii) $\Delta(x) = \Delta(x) + x$

(iii) $\Delta(x) = \tau\cdot\Delta(x)$

PROOF. (i) is a consequence of TI5: by DE_1, $\Delta(x) = \tau_{\{i\}}(z)$ where $z = iz + x$, i not in x. Also by DE_1, $\Delta(xy) = \tau_{\{i\}}(u)$ where $u = iu + xy$, i not in xy. Now

$$zy = (iz + x)y = izy + xy$$

and therefore $zy = u$ (since both solve the same guarded recursion equation). Hence

$$\tau_{\{i\}}(zy) = \tau_{\{i\}}(z)\cdot y = \Delta(x)\cdot y = \tau_{\{i\}}(u) = \Delta(xy).$$

(ii): $\Delta(x) = \tau_{\{i\}}(y)$ where $y = iy + x$, $i \notin \alpha(x)$. So

$$\Delta(x) = \tau_{\{i\}}(iy + x) = \tau\cdot\tau_{\{i\}}(y) + x = \tau\Delta(x) + x =$$

$$\tau\Delta(x) + x + x = \Delta(x) + x.$$

(iii): As in (ii), $\Delta x = \tau\Delta x + x \underset{(ii)}{=} \tau(\Delta x + x) + x \underset{T1-3}{=} \tau(\Delta x + x) = \tau\Delta x.$ □

1.1.2.1. REMARK. An inaccuracy in the above proof is the assumption that a fresh i for process x can be found; if x uses the whole alphabet this is not possible. We will not be bothered by this fact, and adopt (i)-(iii) of the Proposition above for all processes.

1.1.3. NOTATION. By means of the ad hoc notation τ^ω the process denoted by $\tau_{\{i\}}(x)$ where $x = ix$ is meant. So DE_1 now yields:

$$\boxed{\frac{x = ix = ix + \delta}{\tau^\omega = \tau_{\{i\}}(x) = \Delta\tau_{\{i\}}(\delta) = \Delta\delta}}$$

We will call τ^ω (or $\Delta\delta$) henceforth: *livelock.*

1.1.4. NOTATION. Henceforth we will write Δx instead of $\Delta(x)$; the resulting ambiguity in Δxy is harmless in view of Proposition 1.1.2(i).

1.2. *A model with bisimulation semantics and explicit divergence.*

We suppose as known the definition in BAETEN, BERGSTRA & KLOP [2] of $\mathcal{G}_{\aleph_1}$, the set of *countably branching process graphs*, as well as the definition of the operations + and $\cdot$ on such graphs. In the present situation we have as only extra definition of the syntax interpretation that for $g \in \mathcal{G}_{\aleph_1}$, $\Delta(g)$ is g with a τ-loop at the root. The intended model of BS_Δ will be

$$\mathcal{G}_{\aleph_1} / \underline{\leftrightarrow}_{r\tau\Delta}$$

where $\underline{\leftrightarrow}_{r\tau\Delta}$ is $r\tau\Delta$-bisimilarity, which we are going to define in a number of steps. In the intended model, *no abstraction from divergence is possible*; in other words, the model keeps track of the divergence possibilities in a process.

1.2.1. *Divergence preserving bisimulations.*

1.2.1.1. DEFINITION. Let $g \in \mathcal{G}_{\aleph_1}$.

(i) Steps $s \xrightarrow{u} t$ and $s \xrightarrow{v} t'$ (where $u,v \in A \cup \{\tau\} \cup \{\delta\}$; s,t,t' are nodes of g) are *brothers*. The step $t \xrightarrow{v} t'$ is a *son* of the step $s \xrightarrow{u} t$.

(ii) g is said to be *δ-normalised* if δ-steps have no brothers and no sons. (Here a δ-step has the form $o \xrightarrow{\delta} o$.)

(iii) Endpoints of δ-steps are *virtual* nodes; all other nodes in g are *proper*.

(iv) A node is a *deadlock node* if all outgoing traces have only edges with labels τ,δ and end all in δ. (See Figure 2.)

(v) Nodes from which only infinite τ-traces start, are *livelock* nodes.

(vi) A *deadlock/livelock node* is a node from which all outgoing traces have as labels only τ,δ and such that there is no succesfully terminating trace.

(vii) Nodes from which an infinite τ-trace starts, are *divergence* nodes. (In particular, a livelock node is a divergence node.)

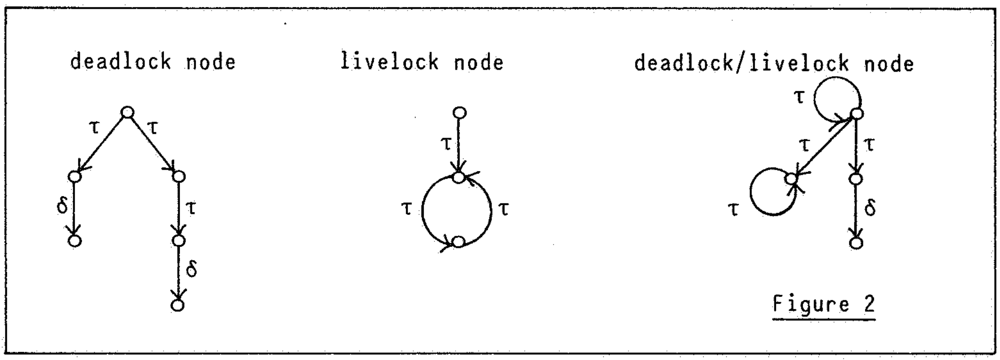


Figure 2

1.2.1.2. DEFINITION. A *path* π in g is a sequence

$$s_0 \xrightarrow{u_0} s_1 \xrightarrow{u_1} \dots \xrightarrow{u_{n-1}} s_n \qquad (n \geqslant 0)$$

of proper nodes and labeled edges. The node s_0 is *begin*(π), the node s_n is *end*(π). The path π determines a sequence of labels $u_0u_1\dots u_{n-1}$ $(u_i \in A \cup \{\tau\})$; *val*$(\pi)$ is this sequence with all τ's skipped.
Note that *val*$(\pi) \in A^*$, the set of words over A, including the empty word λ.

1.2.1.3. DEFINITION. Let $g,h \in \mathcal{G}_{\aleph_1}$ be δ-normalised. Let R be a relation between the proper nodes of g,h. We say that R *relates* path π in g to path π' in h (notation π R π') if

begin(π) R *begin*(π')
end(π) R *end*(π')
val(π) = *val*(π').

(s R t means: nodes s,t are related by R.) If π R π', we also say that π is *transfered* by R to π', and vice versa.

(ii) Relation R has the *transfer property* if whenever $\pi \in g$ and $\mathit{begin}(\pi)$ R t, $t \in h$, then π is transfered to some $\pi' \in h$ with $\mathit{begin}(\pi') = t$. Likewise with the role of g,h interchanged. (Note that by definition the end-points of π, π' are again related!)

1.2.1.4. <u>DEFINITION</u>. (i) Let $g,h \in \mathcal{G}_{\aleph_1}$ be in δ-normal form. Then

$$g \overset{R}{\underline{\leftrightarrow}}_{r\tau\Delta} h$$

(g,h are $r\tau\Delta$-bisimilar via R) if there is a relation between the proper nodes of g,h such that

(1) the roots of g,h are related

(2) a root may only be related to a root

(3) R has the transfer property

(4) a deadlock node may only be related to a deadlock node

(5) a divergence node may only be related to a divergence node.

(ii) $g \underline{\leftrightarrow}_{r\tau\Delta} h$ if $g \overset{R}{\underline{\leftrightarrow}}_{r\tau\Delta} h$ for some R.

For use in the next section we also define $r\tau$-bisimilarity:

1.2.1.5. <u>DEFINITION</u>. (i) Let $g,h \in \mathcal{G}_{\aleph_1}$ be in δ-normal form. Then

$$g \overset{R}{\underline{\leftrightarrow}}_{r\tau} h$$

(g,h are $r\tau$-bisimilar via R) if

(1)-(3) as in Definition 1.2.1.4, and

(4)': a deadlock/livelock node may only be related to a similar node.

(ii) $g \underline{\leftrightarrow}_{r\tau} h$ if there is an R such that $g \overset{R}{\underline{\leftrightarrow}}_{r\tau} h$.

For a rephrased but equivalent definition of $\underline{\leftrightarrow}_{r\tau}$ see [1].

Another equivalent definition is obtained by replacing (4)' by:

(5)': a node with possibly succesful termination may only be related to a similar node. (Here a node has 'possibly succesful termination' if there is an outgoing trace ending succesfully.)

1.2.1.6. REMARK. Clause (2) in Definitions 1.2.1.4 and 1.2.1.5 has not so much an intuitive justification, but rather a formal (algebraical) justification: the resulting notion of bisimilarity is then a congruence.

1.2.1.7. EXAMPLE.

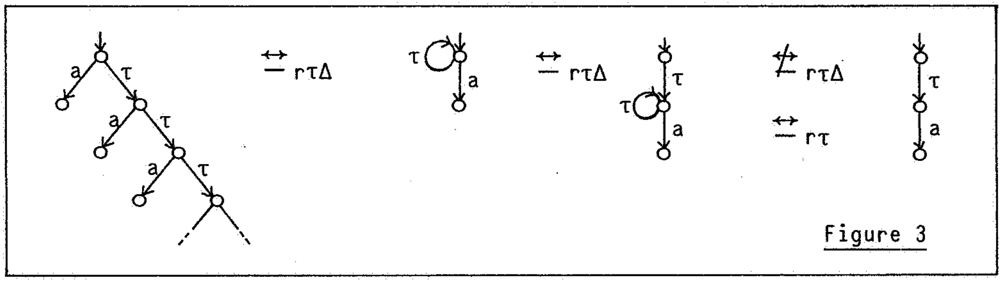

a
τ
a
τ
a
τ
$\underline{\leftrightarrow}_{r\tau\Delta}$
τ
a
$\underline{\leftrightarrow}_{r\tau\Delta}$
τ
τ
a
$\not\underline{\leftrightarrow}_{r\tau\Delta}$
$\underline{\leftrightarrow}_{r\tau}$
τ
a

Figure 3

We omit the routine proof of the following theorem.

1.2.1.8. THEOREM. (i) $\underline{\leftrightarrow}_{r\tau\Delta}$ *is a congruence w.r.t. the operations* + *and* • *in the graph domain* $\mathcal{G}_{\aleph_1}$.

(ii) $\mathcal{G}_{\aleph_1}/\underline{\leftrightarrow}_{r\tau\Delta} \models BS_\Delta$. □

1.2.1.9. NOTATION. Henceforth $\mathcal{G}_{\aleph_1}/\underline{\leftrightarrow}_{r\tau\Delta}$ will be called $\mathcal{A}(BS_\Delta)$.

1.2.1.10. REMARK. Note that in the model $\mathcal{A}(BS_\Delta)$ the delay operator is idempotent: $\Delta\Delta x = \Delta x$.

1.3. *Extension with the empty process* ε.

A very useful constant for process algebra is the empty process ε, subject to the axioms

$$x\varepsilon = x$$

$$\varepsilon x = x$$

$$\tau_I(\varepsilon) = \varepsilon$$

(which are to be added, when adopting ε, to BS_Δ in Table 1). It can be consistently added to BS_Δ -cf. KOYMANS & VRANCKEN [16]. We have anticipated working with ε by formulating the first τ-law in Table 1 as aτ = a, ττ = τ; otherwise we could have taken xτ = x, but this does not hold for ε, as it would lead to the 'inconsistency' ετ = ε = τ. (See Example 1.4.4(vi) below.)

A positive effect of ε is that in its presence Δ can be viewed as a *constant* rather than as an operator. In the algebra Δ could be introduced as a constant rightaway, but in the model $\mathcal{A}(BS_\Delta)$ there is no interpretation for Δ as a constant. However, with the addition of ε (which is then also admitted as a label for edges of process graphs in $\mathcal{G}_{\aleph_1}$), Δ has as interpretation the process graph

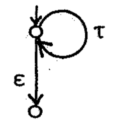


Figure 4

Indeed, since $\Delta\varepsilon = \Delta$, we have by DE_1:

$$\Delta = \tau_{\{i\}}(x) \text{ where } x = ix + \varepsilon.$$

It is even possible to admit infinitely long ε-traces in the process graphs from which a model of

$$BS_\Delta^\varepsilon = BS_\Delta + \{x\varepsilon = x,\ \varepsilon x = x,\ \tau_I(\varepsilon) = \varepsilon\}$$

is obtained. The proper definition of bisimulation which also works for ε, in absence of Δ, is given and studied in KOYMANS & VRANCKEN [16]. Here we will only mention this possibility of the enrichment with ε; for the failure semantics axiomatisations in the sequel, in which we are primarily interested in this paper, we will be more explicit about ε.

Note that the possibility of introducing ε gives us $(\Delta x)y = \Delta(xy)$ at once as a consequence of the associativity of $\cdot$; this is in accordance with Proposition 1.1.2 and Notation 1.1.4.

Also note that some of the axioms in Table 1 can be simplified using ε: $x + x = x$ is equivalent with $\varepsilon + \varepsilon = \varepsilon$, since

$$\varepsilon + \varepsilon = \varepsilon \quad\Longrightarrow\quad (\varepsilon + \varepsilon)x = \varepsilon x \quad\Longrightarrow\quad \varepsilon x + \varepsilon x = \varepsilon x \quad\Longrightarrow\quad x + x = x$$

and likewise

$$x + \delta = x$$

$$\tau x + x = \tau x$$

$$\tau_I(\Delta x) = \Delta(\tau_I(x))$$

are equivalent with respectively

$$\varepsilon + \delta = \varepsilon$$
$$\tau + \varepsilon = \tau$$
$$\tau_I(\Delta) = \Delta.$$

1.4. *Consistency of process axiomatisations.*

In this paper we will adopt the following criterion for consistency of process axiomatisations: if T is such an axiomatisation (e.g. BS_Δ in Table 1 above), then whenever T yields equality of two finite and closed process expressions p,q it should be the case that the sets of complete traces, *trace*(p) and *trace*(q), coincide. Here a 'complete trace' is a trace ending succesfully ($\sigma \in A^*$) or unsuccesfully ($\sigma\delta$, $\sigma \in A^*$); τ and ε do not appear in σ.

1.4.1. NOTATION. BPA (basic process algebra), BPA_δ, $BPA_{\delta\varepsilon}$, and $BPA_{\delta\varepsilon\tau}$ are the subsystems of BS_Δ^ε as displayed in Table 2:

TABLE 2. $BPA_{\delta\varepsilon\tau}$, basic process algebra with δ,ε,τ

	$x + y = y + x$	A1
	$(x + y) + z = x + (y + z)$	A2
	$x + x = x$	A3
	$(x + y)z = xz + yz$	A4
BPA	$(xy)z = x(yz)$	A5
	$\delta + x = x$	A6
BPA_δ	$\delta x = \delta$	A7
	$\varepsilon x = x$	E1
$BPA_{\delta\varepsilon}$	$x\varepsilon = x$	E2
	$a\tau = a$	T1
	$\tau\tau = \tau$	T1'
	$\tau x + x = \tau x$	T2
$BPA_{\delta\varepsilon\tau}$	$a(\tau x + y) = a(\tau x + y) + ax$	T3

1.4.2. DEFINITION. Let t be a closed term built from $\delta,\varepsilon,\tau,a,b,c,\ldots$ and $+,\cdot$. We say that t is in (BPA$_{\delta\varepsilon\tau}$-) *normal form* if in t no rewritings are possible of the form (see Table 3):

TABLE 3

$(x+y)z$	$\longrightarrow$	$xz+yz$
$\delta+x$	$\longrightarrow$	x
δx	$\longrightarrow$	δ
εx	$\longrightarrow$	x
$x\varepsilon$	$\longrightarrow$	x
$a\tau$	$\longrightarrow$	a
$\tau\tau$	$\longrightarrow$	τ

Note that subterms of a term in normal form are again in normal form. This makes the following inductive definition possible of *trace*(t), for t a closed normal form:

1.4.3. DEFINITION. Let t be a closed normal form. Then *trace*(t) is defined by the following clauses in Table 4:

TABLE 4

$\mathit{trace}(\varepsilon)$	$=$	$\emptyset$
$\mathit{trace}(\tau)$	$=$	$\emptyset$
$\mathit{trace}(\delta)$	$=$	$\{\delta\}$
$\mathit{trace}(a)$	$=$	$\{a\}$
$\mathit{trace}(t+s)$	$=$	$\mathit{trace}(t) \cup \mathit{trace}(s)$
$\mathit{trace}(a\cdot t)$	$=$	$\{a\sigma \mid \sigma \in \mathit{trace}(t)\}$
$\mathit{trace}(\varepsilon\cdot t)$	$=$	$\mathit{trace}(t)$
$\mathit{trace}(\tau\cdot t)$	$=$	$\mathit{trace}(t)$

1.4.4. DEFINITION. Let T be a process axiomatisation. Let $\overline{T}$ be the set of closed equations derivable from T. Then T is called *trace inconsistent* if $\overline{T}$ contains an equation $t_1 = t_2$, where t_1,t_2 are closed BPA$_{\delta\varepsilon\tau}$-terms, such that $\mathit{trace}(t_1) \neq \mathit{trace}(t_2)$. Otherwise T is called *trace consistent*.

1.4.5. EXAMPLE. (i) $BPA \cup \{z(x+y) = zx+zy\}$ is trace consistent.

(ii) $BPA_\delta \cup \{z(x+y) = zx+zy\}$, on the other hand, is trace inconsistent, since the consequence $a(b+\delta) = ab+a\delta$ equates expressions with different trace sets (viz. $\{ab\}$ resp. $\{ab, a\delta\}$).

(iii) $BPA_\delta \cup$ the single instance $a(b+c) = ab+ac$ is trace consistent.

(iv) BPA_δ extended with $a(b+c) = ab+ac$ plus renaming operators a_H (as in BERGSTRA, KLOP & OLDEROG [8]) is inconsistent.

(v) ACP plus $a(b+c) = ab+ac$ is trace inconsistent (for a suitable communication function). For ACP, see [3,4].

(vi) $BPA_{\delta\varepsilon\tau} \cup \{\varepsilon = \tau\}$ is trace inconsistent, since from it we derive:

$$a+\tau\delta = a+\varepsilon\delta = a+\delta = a$$

and $\mathit{trace}(a+\tau\delta) = \{a,\delta\} \neq \mathit{trace}(a) = \{a\}$.

1.4.6. REMARK. (i) Note that trace equality in the above sense is not a congruence; while $\mathit{trace}(\varepsilon) = \mathit{trace}(\tau)$ it is not the case that $\mathit{trace}(a+\tau\delta) = \mathit{trace}(a+\varepsilon\delta)$.

(ii) The present definition of trace inconsistency, referring to unwanted identifications of *finite* processes, is sufficient for our purpose in this paper. When infinite processes are considered, a 'right' notion of trace inconsistency is not so immediate as for finite processes. Of course, identifying a^ω and b^ω would certainly be considered as trace inconsistent, but w.r.t. succesful termination / livelock / deadlock the situation is more difficult. E.g. the equation $ab = ab+a\tau^\omega$ holds in the failure semantics FS of Section 4 below; the equation $a\tau^\omega = a\delta$ holds in bisimulation semantics BS of Section 2. (Hence, FS + BS is trace inconsistent as it derives $ab = ab+a\delta$.) We will not attempt a formulation of trace inconsistency which takes also infinite processes into account. Note, nevertheless, that the present definition (1.4.4) also works in the presence of infinite processes: for, although we look at inconsistencies induced at the level of finite processes, these inconsistencies may be derived using detours in which infinite processes occur.

2. BISIMULATION SEMANTICS WITH FAIR ABSTRACTION: BS

The first extension of the kernel axiom system BS_Δ introduced in Section 1 consists of the simple addition

$$\Delta = \tau.$$

In fact, Δ being an operator, we should write $\Delta(x) = \tau \cdot x$; but as we have seen in Section 1.3, it is consistent to think of Δ as a process itself, when ε is introduced. We will denote $BS_\Delta \cup \{\Delta = \tau\}$ by BS.

The delay rule DE now reads:

$$\boxed{\text{KFAR} \quad \frac{(\forall k \in \mathbb{Z}_n) \quad x_k = i_k \cdot x_{k+1} + y_k \quad , \quad i_k \in I}{\tau_I(x_0) = \tau \cdot \tau_I(\sum y_k)}}$$

where KFAR is Koomen's Fair Abstraction Rule. An extension of this axiom system BS, namely ACP_τ + KFAR + some other proof principles, was studied extensively in BAETEN, BERGSTRA & KLOP [1]. The adjective 'fair' in KFAR denotes the presence of a fair scheduler of steps: eventually a step not on the i_0-i_1-...-i_{k-1} cycle in the premiss of KFAR will be taken.

An application of a protocol verification using KFAR can be found in BERGSTRA & KLOP [5].

A model with bisimulation semantics with fair abstraction is studied in [1]. It is similar to the model $\mathcal{A}(BS_\Delta)$ appearing in Theorem 1.2.1.8: instead of $r\tau\Delta$-bisimilarity we divide out the coarser congruence $r\tau$-bisimilarity defined in Definition 1.2.1.5. In the notation of this paper this yields the model $\mathcal{A}$(BS) and we have $\mathcal{A}(BS) \models BS$. Properties of $\mathcal{A}$(BS) are studied in [1]. In this paper we will not be further concerned with $\mathcal{A}$(BS).

We note the following important fact, viz. that in this semantics BS livelock (= τ^ω, or $\Delta\delta$) and deadlock are identified:

2.1. <u>PROPOSITION</u>. BS $\vdash \Delta\delta = \tau\delta$.

<u>PROOF</u>. $\Delta = \tau$, hence $\Delta\delta = \tau\delta$. □

3. FAILURE SEMANTICS WITH EXPLICIT DIVERGENCE: FS_Δ

We will now turn to the second extension of the kernel system BS_Δ, namely by adding axioms which lead us to failure semantics. We will do this in three steps: first for finite processes in absence of τ-steps, then for finite processes with τ-steps and finally for general processes with τ-steps.

3.1. *Failure semantics for finite processes without silent steps.*

This is the subject of BERGSTRA, KLOP & OLDEROG [8], where it turns out that a *complete* axiom system (complete w.r.t. the usual failure semantics definition for finite processes; see below) is obtained by addition to BPA_δ (as in Table 2) of the following axioms:

$$a(bx+u) + a(by+v) = a(bx+by+u) + a(bx+by+v) \qquad R1$$

$$a(b+u) + a(by+v) = a(b+by+u) + a(b+by+v) \qquad R2$$

$$ax + a(y+z) = ax + a(y+z) + a(x+y) \qquad S$$

These axioms are very much inspired by the axiom system in BROOKES [9]; there silent moves are present but the axioms there imply R1,2 and S. Usually axiom S (S for saturation) is seen in the literature in the form of the equivalent pair of 'convexity' axioms (see e.g. BROOKES [9], DE NICOLA & HENNESSY [21]):

$$ax + ay = ax + ay + a(x+y) \qquad CO1$$

$$ax + a(x+y+z) = ax + a(x+y) + a(x+y+z) \qquad CO2$$

In fact, axioms R2 and S will become superfluous after τ-steps plus corresponding axioms for failure semantics are introduced, as in 3.2 below. (In the presence of ε the axiom R2 is already here derivable.)

A useful consequence of R1,2 (R for 'readiness'; without S the axioms R1,2, when added to BPA_δ, completely describe readiness semantics - see [8]) is obtained by taking $u = v = \delta$:

$$abx + aby = a(bx+by) \qquad R3$$

$$ab + aby = a(b+by) \qquad R4$$

Here a,b range over $A_{\delta\tau}$. (*Warning*: a,b may not be ε; e.g. R3 yields an inconsistency if $b = \varepsilon$.)

As shown in BERGSTRA, KLOP & OLDEROG [8], the initial algebra of BPA_δ + R1,2 + S is isomorphic to the domain $\mathbb{H}$ of finite acyclic process graphs modulo failure equivalence ($\equiv_F$). We will explain $\equiv_F$, which is in fact a congruence, now. First we fix some notation and preliminary concepts.

3.1.1. DEFINITION. (i) $\mathbb{H}$ is the subset of $\mathcal{G}_{\aleph_1}$ consisting of finite, nonzero process graphs without cycles and with edges labeled by atoms from $A_\delta = A \cup \{\delta\}$. Moreover, the graphs are supposed to be δ-normalised (cf. Definition 1.2.1.1).

(ii) $\mathbb{H}^\circ = \mathbb{H} \cup \{\mathbb{O}\}$ where $\mathbb{O}$ $(= \downarrow\circ)$ is the zero graph consisting of one node only. (The zero graph is the interpretation of the empty process ε.)

3.1.2. DEFINITION. Let $g,h \in \mathbb{H}^\circ$ and $\sigma \in A^*$, the set of words over A (not A_δ). The empty word in A^* is λ. We say that

$$g \xrightarrow{\sigma} h$$

is a *derivation* (from g to h via σ) if h is a subgraph of g such that the root of h is reachable from the root of g via a path whose labels form the word σ.

3.1.3. EXAMPLE. See Figure 5.

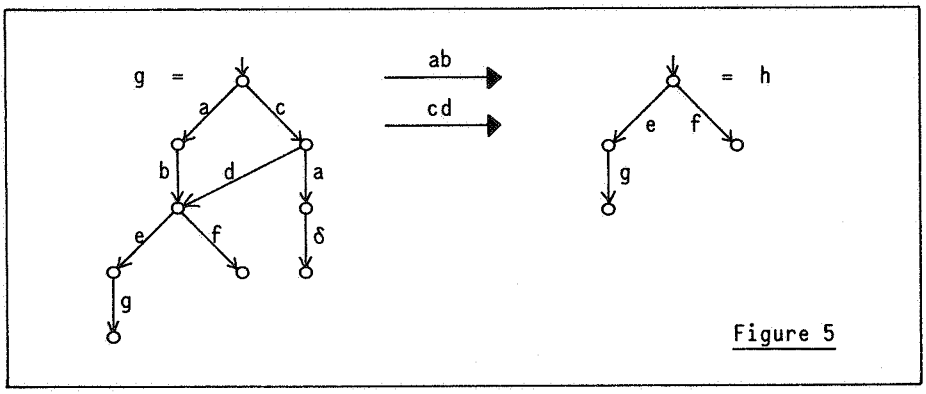


Figure 5

We also have in this example $g \xrightarrow{abf} 0$, $g \xrightarrow{ca} \delta$ (where δ is $\rightarrow\circ\xrightarrow{\delta}\circ$); but *not* $g \xrightarrow{ca\delta} 0$.

3.1.4. DEFINITION. (i) Let $g \in \mathbb{H}$, $g \neq \delta$ and $g \neq 0$. Then $I(g)$ is the set of *initial* steps of g.

(ii) $I(\delta) = \emptyset$

(iii) (notation:) if $X \subseteq A$, then $\overline{X} = A - X$.

3.1.5. DEFINITION. $\mathcal{F}[\![g]\!]$, the *failure set* of g, is defined as the least set obtained by the following clauses (see Table 5). For all $\sigma \in A^*$:

TABLE 5. Failure set for finite processes without τ

(i)	$g \xrightarrow{\sigma} h \neq \mathbb{0}$	$\Rightarrow$	$(\sigma, \overline{I(h)}) \in \mathcal{F}[\![g]\!]$
(ii)	$g \xrightarrow{\sigma} \mathbb{0}$	$\Rightarrow$	$\sigma \in \mathcal{F}[\![g]\!]$
(iii)	$(\sigma, X \cup Y) \in \mathcal{F}[\![g]\!]$	$\Rightarrow$	$(\sigma, Y) \in \mathcal{F}[\![g]\!]$

Note that $\mathcal{F}[\![g]\!]$, next to *failure pairs* (σ, X) (in [8] written as $[\sigma, X]$), contains traces σ for succesful traces. This definition of the failure set is taken from BERGSTRA, KLOP & OLDEROG [8].

3.1.6. DEFINITION. $g \equiv_{\mathcal{F}} h$ iff $\mathcal{F}[\![g]\!] = \mathcal{F}[\![h]\!]$,

in words: g,h are *failure equivalent*.

The present definition does not lend itself for an easy extension to the case where silent steps are present, as in the next subsection 3.2. The problem is that we wish to distinguish e.g. graphs corresponding to $\tau + a$ and $\tau + \tau a$. A straightforward extension of the definition in Table 5 (namely by only extending the definition of $I(g)$ in the obvious way) would yield as failure set for both graphs:

$$\{\lambda,\ (\lambda, \overline{\{a\}}),\ a\}.$$

Therefore we amend the definition in 3.1.5, without altering the failure equivalence, by using an auxiliary step ε which is to be appended after each succesful trace in g. Indeed, this is consistent with our consideration of the empty process ε. (Cf. BROOKES, HOARE & ROSCOE [10] where ✓ is used to denote succesful termination.)

3.1.7. DEFINITION. (i) $A_\varepsilon = A \cup \{\varepsilon\}$, (ii) $I(\mathbb{O}) = \{\varepsilon\}$.

(The definition of $I(g)$ for g not δ or $\mathbb{O}$ is as in 3.1.4.)

(iii) If $X \subseteq A$, then $\overline{X} = A_\varepsilon - X$.

3.1.8. DEFINITION. The amended failure set of g, $\mathcal{F}'[\![g]\!]$, is defined thus: for all $\sigma \in A^*$

TABLE 6. Failure set for finite processes without τ

(i)	$g \xrightarrow{\sigma} h$	$\Rightarrow$	$(\sigma, \overline{I(h)}) \in \mathcal{F}'[\![g]\!]$
(ii)	$g \xrightarrow{\sigma} \mathbb{O}$	$\Rightarrow$	$\sigma\varepsilon \in \mathcal{F}'[\![g]\!]$
(iii)	$(\sigma, X \cup Y) \in \mathcal{F}'[\![g]\!]$	$\Rightarrow$	$(\sigma, Y) \in \mathcal{F}'[\![g]\!]$

Note that a succesful termination node now gives two contributions: $(\sigma, \overline{\{\varepsilon\}})$ as well as $\sigma\varepsilon$. Again we define: $g \equiv_{\mathcal{F}'} h$ iff the amended failure sets coincide. The proof of the proposition that $\equiv_{\mathcal{F}'}$ coincides with $\equiv_{\mathcal{F}}$ is left to the reader. Henceforth we will write $\mathcal{F}[\![g]\!]$ instead of $\mathcal{F}'[\![g]\!]$, and we will not use the definition in Table 5 anymore.

3.1.9. NOTATION. X^c is the least set containing X such that

$$(\sigma, X \cup Y) \in X \quad \Rightarrow \quad (\sigma, X) \in X^c.$$

3.1.10. EXAMPLE. (i) $\mathcal{F}[\![\delta]\!] = \{(\lambda, A_\varepsilon)\}^c$

(ii) $\mathcal{F}[\![a]\!] = \{(\lambda, \overline{\{a\}}), (a, \overline{\{\varepsilon\}}), a\varepsilon\}^c$

(iii) $\mathcal{F}[\![a + a\delta]\!] = \{(\lambda, \overline{\{a\}}), (a, \overline{\{\varepsilon\}}), a\varepsilon, (a, A_\varepsilon)\}^c$

(Here 'a' is the graph consisting of one a-step, etc.)

It is proved in BERGSTRA, KLOP & OLDEROG [8] that failure equivalence is a congruence on $\mathbb{H}$ w.r.t. + and $\cdot$, and that we have the isomorphism

$$\mathbb{H}/\equiv_{\mathcal{F}} \;\cong\; I(BPA_\delta + R1,2 + S)$$

where the RHS denotes the initial algebra of the axiom system in question. Furthermore, [8] mentions the isomorphism

$$\mathbb{H}/\underline{\leftrightarrow}_\delta \;\cong\; I(BPA_\delta)$$

where $\underline{\leftrightarrow}_\delta$ is bisimulation congruence.

3.2. *Bisimulation and failure semantics for finite processes with silent steps.*

We will now extend the situation of Section 3.1 by admitting silent steps or τ-steps. Essentially, the results in this section 3.2 are from BROOKES [9]; the difference is that we strictly adhere to the use of *congruences* and, therefore, have the definitions of bisimulation and failure equivalence slightly adapted as compared to the similar notions in Brookes' work. The results are displayed in Table 7.

TABLE 7. An axiomatisation of bisimulation and failure semantics for finite processes with $+,\cdot,\delta$ and τ.

FAILURE SEMANTICS			
BISIMULATION SEMANTICS			
$x+y = y+x$	A1		
$(x+y)+z = x+(y+z)$	A2		
$x+x = x$	A3		
$(x+y)z = xz+yz$	A4		
$(xy)z = x(yz)$	A5		
$\delta+x = x$	A6		
$\delta x = \delta$	A7		
		$a(bx+u)+a(by+v) = a(bx+by+u)+a(bx+by+v)$	R1
		$a(b+u)+a(by+v) = a(b+by+u)+a(b+by+v)$	R2
		$ax+a(y+z) = ax+a(y+z)+a(x+y)$	S
	$\mathbb{H}_\delta/\underline{\leftrightarrow}_\delta$		$\mathbb{H}_\delta/\equiv_F$
$a\tau = a$	T1		
$\tau\tau = \tau$	T1'		
$\tau x+x = \tau x$	T2		
$a(\tau x+y) = a(\tau x+y)+ax$	T3		
		$\tau x+y = \tau x+\tau(x+y)$	T4
	$\mathbb{H}_\delta/\underline{\leftrightarrow}_{r\tau\delta}$		$\mathbb{H}_\delta/\equiv_F$ τ-case

The axioms between brackets are in fact derivable, as we will show below. Further, 'a' in Table 7 varies over $A_{\delta\tau} = A_\delta \cup \{\tau\}$. The axioms T1-3 are Milner's τ-laws, introduced in MILNER [18]; $\underline{\leftrightarrow}_{r\tau\delta}$ is the notion of bisimulation suitable to deal with τ-steps, called rooted τδ-bisimulation, introduced in BERGSTRA & KLOP [3]. Except for the fact that in the present setting traces end either succesfully or in δ, this notion of bisimulation is the *observational congruence* in MILNER [18]. The adjective 'rooted' refers to the restriction imposed on a rτδ-bisimulation that a root may only be related to a root (see Definition 1.2.1.5(ii)); this is just what is needed to obtain the congruence property. Without it, one obtains the well-known fact that

$$a \leftrightarrows_{\tau\delta} \tau a$$

while

$$a+b \not\leftrightarrows_{\tau\delta} \tau a+b.$$

Here $\leftrightarrows_{\tau\delta}$ corresponds to Milner's observational equivalence.

To extend failure semantics to the case where silent steps are present, the alphabet is extended with τ, result: $A_{\delta\tau}$. The process graph domain $\mathbb{H}$ now consists of finite acyclic nonzero process graphs with edges labeled by elements of $A_{\delta\tau}$; again the graphs are supposed to be δ-normalised. Definition 3.1.2 of 'derivation' is extended as follows:

3.2.1. DEFINITION. Let $g,h \in \mathbb{H}^\circ$ (= $\mathbb{H} \cup \{\mathbb{0}\}$) and let $\sigma \in A^*$. Then $g \xrightarrow{\sigma} h$ if h is a subgraph of g such that the root of h can be reached from the root of g via a path whose labels yield, *after omitting τ-labels*, $\sigma \in A^*$.

3.2.2. DEFINITION. The set I(g) of initial steps of g contains those steps in A which can be reached from the root of g via a path with label λ, i.e. a path consisting of zero or more silent steps.

A concept which will play an important role in the sequel, is that of an 'unstable' process. This is a process obtained from an unstable process graph:

3.2.3. DEFINITION. Let $g \in \mathcal{G}_{\aleph_1}$. Then g is *unstable* if g has an initial τ-step.

The notion of unstable processes was introduced in MILNER [18].

We can now define the failure set of a process graph in $\mathbb{H}$:

3.2.4. DEFINITION. The failure set $F[\![g]\!]$, for $g \in \mathbb{H}$, is defined as the least set satisfying the following clauses (in Table 8). For all $\sigma \in A^*$:

TABLE 8. Failure set for finite processes with τ

(i)	$g \xrightarrow{\sigma} h$	$\Rightarrow$	$(\sigma, \overline{I(h)}) \in \mathcal{F}[\![g]\!]$
(ii)	$g \xrightarrow{\sigma} 0$	$\Rightarrow$	$\sigma\varepsilon \in \mathcal{F}[\![g]\!]$
(iii)	$(\sigma, X \cup Y) \in \mathcal{F}[\![g]\!]$	$\Rightarrow$	$(\sigma, X) \in \mathcal{F}[\![g]\!]$
(iv)	g is unstable	$\Rightarrow$	$\tau \in \mathcal{F}[\![g]\!]$

Here clauses (i)-(iii) are as before in Definition 3.1.8, except for the more general definition of 'initial step'. Again we define g,h to be failure equivalent if $\mathcal{F}[\![g]\!] = \mathcal{F}[\![h]\!]$, notation $g \equiv_{\mathcal{F}} h$.

Without the extra clause (iv), the resulting failure equivalence would not be a congruence, for the same reason as mentioned above for the case of bisimulation. The clause (iv) merely serves to distinguish stable process graphs from unstable, and this is sufficient to have a congruence.

3.2.5. EXAMPLE. $\tau a \not\equiv_{\mathcal{F}} a$, since $\mathcal{F}[\![\tau a]\!] = \{(\lambda, \overline{\{a\}}), (a, \overline{\{\varepsilon\}}), a\varepsilon, \tau\}^{c}$ and $\mathcal{F}[\![a]\!] = \mathcal{F}[\![\tau a]\!] - \{\tau\}$.

As indicated in Table 7, the axioms in the left column have an initial algebra which is isomorphic to $\mathbb{H}/\underline{\leftrightarrow}_{r\tau\delta}$. This fact is mentioned in BERGSTRA, KLOP & OLDEROG [8]. The initial algebra of all axioms in Table 7 is isomorphic to $\mathbb{H}/\equiv_{\mathcal{F}}$; essentially this is proved in BROOKES [9]. Taking care of some minor differences between the present definitions and those of BROOKES [9] is left to the reader. One such difference is caused by our separation of stable and unstable processes (by clause (iv) in Table 8), another one is due to the possible presence in our setting of 'silent exits', that is, a subprocess of the form $\tau + x$. (Such subprocesses do not occur in Brookes' framework since every branch ends there in NIL.)

We conclude this subsection about finite processes with some simple observations.

In the algebra, a process x is called unstable iff it is of the form $\tau y + z$ for some y,z. Now:

3.2.6. PROPOSITION. *In failure semantics:* x *is unstable iff* $x = \tau x$.

PROOF. $x = \tau y + z = \tau(y+z) + \tau y = \tau\tau(y+z) + \tau\tau y = \tau(\tau(y+z) + \tau y) =$

$\tau(\tau y + z) = \tau x$. □

Our next observation states that in failure semantics "almost all" occurrences of τ, in a finite process, can be eliminated. The proof of this fact is an instructive exercise and hence left to the reader.

3.2.7. PROPOSITION. *Let* $\vdash$ *refer to the axiom system in Table 7. Let* t *be a term such that every 'branch' ends in* δ *or some* $a \in A$ *(not* τ*). Then:*

(i) if t *is stable, there is a* τ*-free* t' *such that* $\vdash t = t'$

(ii) if t *is unstable, there are* τ*-free* $t_1,\ldots,t_n$ *(*$n \geq 1$*) such that*
$\vdash t = \tau t_1 + \ldots + \tau t_n$. □

3.2.8. REMARK. (i) Note that the condition on t is necessary when ε is not present: e.g. $\tau + a$ cannot be proved equal to the form in part (ii) of the proposition.

(ii) On the other hand, when ε is around, (i) and (ii) of the proposition hold without the condition on t. E.g.:

$$\tau + a = a + \tau\varepsilon = \tau(a+\varepsilon) + \tau\varepsilon = \tau(a+\varepsilon) + \tau.$$

3.2.9. PROPOSITION. *The following identities are provable from the axioms in Table 7. Here* $a \in A_{\delta\tau}$.

TABLE 9. Failure semantics for finite processes: derived equations

(i)	$a(\tau x + \tau y)$	$= ax + ay$	
(ii)	$\tau(ax + ay)$	$= \tau ax + \tau ay$	
(iii)	$x + \tau y$	$= \tau(x + \tau y)$	
(iv)	$\tau(ax + ay)$	$= \tau ax + ay$	
(v)	$\tau x + x$	$= \tau x$	(T2 in Table 7)
(vi)	$\tau(x + y)$	$= \tau(x + y) + x$	
(vii)	$a(x + \tau y)$	$= a(x + \tau y) + ay$	(T3 in Table 7)
(viii)	$a(x + \tau y)$	$= a(x + y) + ay$	
(ix)	$a(\tau x + \tau y + \tau z)$	$= ax + ay + az$	
(x)	$ax + ay$	$= ax + ay + a(x + y)$	(C01 in 3.1)
(xi)	$ax + a(x + y + z)$	$= ax + a(x + y) + a(x + y + z)$	(C02 in 3.1)
(xii)	$ax + a(y + z)$	$= ax + a(y + z) + a(x + y)$	(axiom S in 3.1)
(xiii)	$\tau(ax + z) + \tau(ay + z)$	$= \tau(ax + ay + z)$	
(xiv)	$\tau(ax + ay + z)$	$= \tau(ax + z) + ay$	

PROOF. (i) and (ii) follow from R3 in Section 3.1. The identity (iii) is proved in Proposition 3.2.6. Proof of (iv):

$$\tau(ax+ay) \underset{(ii)}{=} \tau ax+\tau ay = \tau ax+\tau ay+\tau ay = \tau(ax+ay)+\tau ay \underset{T4}{=} \tau ax+ay.$$

(T4 is the 'fourth τ-law' in Table 7.)

(v): follows at once from T4.

(vi): $\tau(x+y) \underset{(v)}{=} \tau(x+y)+x+y = \tau(x+y)+x+y+x \underset{(v)}{=} \tau(x+y)+x.$

(vii): $a(x+\tau y) \underset{T4}{=} a(\tau(x+y)+\tau y) \underset{(i)}{=} a(x+y)+ay = a(x+y)+ay+ay = a(x+\tau y)+ay.$

(viii): proved in (vii).

(ix): By (iii), $\tau x+\tau y = \tau(\tau x+\tau y)$. Now:

$$a(\tau x+\tau y+\tau z) \underset{T4}{=} a(\tau(\tau x+\tau y)+\tau z) \underset{(i)}{=} a(\tau x+\tau y)+az \underset{(i)}{=} ax+ay+az.$$

(x): $ax+ay = a(\tau x+\tau y) \underset{(v)}{=} a(\tau x+\tau y+x) \underset{T4}{=} a(\tau x+\tau y+\tau(x+y)) \underset{(ix)}{=} ax+ay+a(x+y).$

(xi):

$$
\begin{aligned}
& ax+a(x+y+z) = \\
& a(\tau x+\tau(x+y+z)) = \ (T4) \\
& a(\tau x+y+z) = \\
& a(\tau x+y+\tau x+y+z) = \ (T4) \\
& a(\tau x+\tau(x+y)+\tau x+\tau(x+y+z)) = \\
& a(\tau x+\tau(x+y)+\tau(x+y+z)) = \\
& ax+a(x+y)+a(x+y+z).
\end{aligned}
$$

(xii):

$$
\begin{aligned}
& ax+a(y+z) = \ (x) \\
& ax+a(y+z)+a(x+y+z) = \ (xi) \\
& ax+a(y+z)+a(x+z)+a(x+y+z) = \ (x) \\
& ax+a(y+z)+a(x+z).
\end{aligned}
$$

(xiii): immediate from R1.

(xiv): $\tau(ax+z)+ay \underset{T4}{=} \tau(ax+ay+z)+\tau(ax+z) \underset{(xiii)}{=} \tau(ax+ay+z).$ □

3.3. *Failure semantics with delay operator.*

In Section 1 we considered an axiom system BS_Δ for bisimulation semantics with explicit divergence; this was an extension of the left column of Table 7. We will now do the same for failure semantics and the resulting axiom system will be an extension of the whole of Table 7: see Table 10. To see that Table 10 extends Table 7, note that some axioms are dropped that have become derivable, and note that DE^∞ generalizes DE_k (the k-periodical versions of DE^∞). Let us first explain this infinitary rule. It says that all exit processes of an infinite τ-trace may be strung together, at the root, by a single τ-loop (see Figure 6). So, as in BS_Δ, a divergence is replaced by a simpler divergence - but still a divergence.

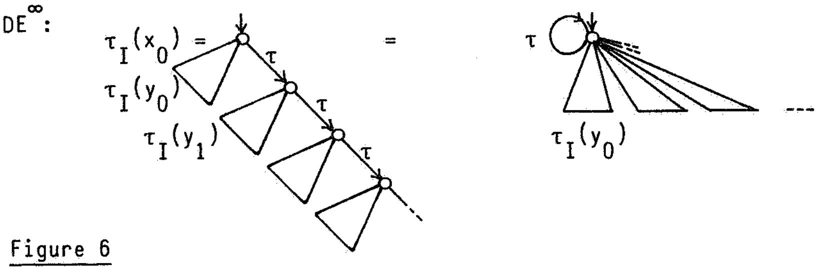


Figure 6

Remarkably, the order in which the exit processes $y_0, y_1, \ldots$ appear along the infinite τ-trace is not important in failure semantics. This can already be seen in the finite case: the order of the exit processes on a τ-trace is lost in failure semantics, as the following example suggests.

3.3.1. EXAMPLE. $\vdash \tau(\tau(\tau(\tau x + a_0) + a_1 + a_2) + a_3 =$

$$\tau(\tau(\tau(\tau x + a_{\pi 0}) + a_{\pi 1}) + a_{\pi 2}) + a_{\pi 3} \qquad (*)$$

where π is any permutation of 0,1,2,3. Here $\vdash$ refers to the axioms for failure semantics in Table 7, which are included in Table 10. The proof of (*) is simple: using $\tau(\tau x + y) = \tau x + y$ we have LHS(*) $= \tau x + a_0 + a_1 + a_2 + a_3$, etc.

The axioms DE1, DE2 are in fact superfluous, as they can be derived from DE^{∞} (in fact even from its consequence DE_1, as shown in Proposition 1.1.2).

Axiom DE4, the *linearity of delay*, is specific for failure semantics. It is an immediate consequence of an equally important property of delay in failure semantics, namely that *prefixing delay amounts to adding livelock*. This is proved in the next proposition.

TABLE 10. Axioms for failure semantics with explicit divergence

$x+y = y+x$	A1
$(x+y)+z = x+(y+z)$	A2
$x+x = x$	A3
$(x+y)z = xz+yz$	A4
$(xy)z = x(yz)$	A5
$\delta+x = x$	A6
$\delta x = \delta$	A7
$a\tau = a$	T1
$\tau\tau = \tau$	T1'
$\tau x+y = \tau x+\tau(x+y)$	T4
$\tau_I(\tau) = \tau$	TI1
$\tau_I(a) = a$ if $a \notin I$	TI2
$\tau_I(a) = \tau$ if $a \in I$	TI3
$\tau_I(x+y) = \tau_I(x)+\tau_I(y)$	TI4
$\tau_I(xy) = \tau_I(x)\cdot\tau_I(y)$	TI5
$\tau_I(\Delta(x)) = \Delta(\tau_I(x))$	TI6
$a(bx+u)+a(by+v) = a(bx+by+u)+a(bx+by+v)$	R1
$\dfrac{\forall k \in \mathbb{N} \quad x_k = i_k \cdot x_{k+1} + y_k \quad (i_k \in I)}{\tau_I(x_0) = \Delta(\sum_k \tau_I(y_k))}$	DE^{∞}
$\Delta x = \tau\Delta x$	DE1
$\Delta x = x+\Delta x$	DE2
$\Delta x = x+\Delta\delta$	DE3
$\Delta(x+y) = \Delta x+\Delta y$	DE4

3.3.2. PROPOSITION. *From* DE^{∞} *the following are derivable:*

(i) $\Delta x = x + \Delta\delta$ (DE3)

(ii) $\Delta(x+y) = \Delta x + \Delta y$ (DE4)

PROOF. Let $i \notin \alpha(x)$, the alphabet used by x. Consider the system of equations

$$\begin{aligned}
z_0 &= i\cdot z_1 + x \\
z_1 &= i\cdot z_2 && (= i\cdot z_2 + \delta) \\
z_2 &= i\cdot z_3 && (= i\cdot z_3 + \delta) \\
&\vdots \\
z_n &= i\cdot z_{n+1} && (= i\cdot z_{n+1} + \delta) \\
&\vdots
\end{aligned}$$

as hypothesis for an application of DE^{∞}, and conclude

(1) $\tau_{\{i\}}(z_0) = \Delta(\tau_{\{i\}}(x+\delta+\delta+\ldots)) = \Delta\tau_{\{i\}}(x) = \Delta x.$

(Here $\tau_{\{i\}}(x) = x$ follows from $i \notin \alpha(x)$.)

On the other hand,

(2) $\tau_{\{i\}}(z_0) = \tau_{\{i\}}(x + i\cdot z_1) = \tau_{\{i\}}(x) + \tau\cdot\tau_{\{i\}}(z_1) = x + \tau\cdot\tau_{\{i\}}(z_1);$

further, an application of DE^{∞} (or already DE_1) yields

(3) $\tau_{\{i\}}(z_1) = \Delta\tau_{\{i\}}(\delta+\delta+\ldots) = \Delta\delta.$

Combining (1),(2),(3) we have

$$\Delta x = x + \tau\Delta\delta = x + \Delta\delta.$$

(ii) follows immediately from (i):

$$\Delta(x+y) = x+y+\Delta\delta = x+\Delta\delta+y+\Delta\delta = \Delta x + \Delta y. \quad \square$$

3.3.3. REMARK. (i) Note that in the proof of Proposition 3.3.2 we have only used the following finitary consequence (let us call it $DE^{[2]}$) of the infinitary rule DE^{∞}:

$$\boxed{\mathrm{DE}^{[2]} \quad \frac{z_0 = i\cdot z_1 + y_0, \qquad z_1 = i\cdot z_1 + y_1}{\tau_{\{i\}}(z_0) = \Delta\tau_{\{i\}}(y_0 + y_1)}}$$

(DE_1, also used in the proof, follows in turn from $\mathrm{DE}^{[2]}$.)
Note also that $\mathrm{DE}^{[2]}$ does not follow from the 'periodical' versions DE_k $(k \geq 1)$.

(ii) In Proposition 3.3.6 an alternative proof of the linearity of Δ is given, which, interestingly, seems quite different and uses the implication

$$\mathrm{R1} + \mathrm{DE}_2 + \mathrm{RSP} \Longrightarrow \Delta(x+y) = \Delta x + \Delta y.$$

Here R1 is the 'readiness' axiom in Table 10, and RSP, the Recursive Specification Principle, states that a guarded system of recursion equations has a unique solution.

This second proof gives the additional information that while BS_Δ is consistent with R1 and also consistent with $\Delta = \tau$, it is inconsistent with $\{\mathrm{R1},\ \Delta = \tau\}$.

(iii) Question: Can $\mathrm{DE}^{[2]}$ be proved from $\mathrm{R1} + \mathrm{DE}_k$ $(k \geq 1) + \mathrm{RSP}$?

3.3.4. REMARK. Pictorially, the proof of Proposition 3.3.2(i) amounts to the following (see Figure 7):

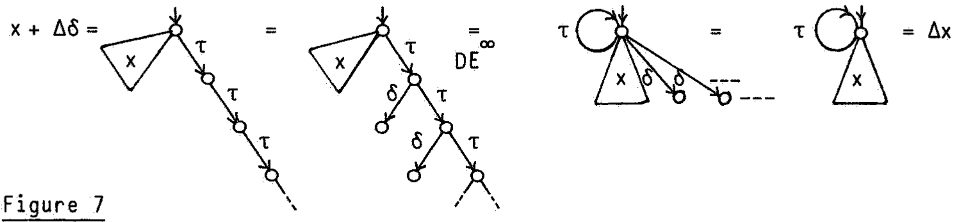


Figure 7

3.3.5. REMARK. In DE^∞ there appears an infinite sum: $\sum_k \tau_I(y_k)$. In the graph model (to be constructed below from $\mathcal{G}_{\aleph_1}$) this is just what is intended (the graphs are countably branching), but in the algebra there are no infinite terms. So, DE^∞ is in fact only applicable in cases where the y_k $(k \geq 1)$ are such that there are only finitely many different $\tau_I(y_k)$.

We will now give a different proof of the linearity of Δ, using the proof principle RSP (Recursive Specification Principle) which, for BS, is studied extensively in BAETEN, BERGSTRA & KLOP [1]. It states that a *guarded system of recursion equations has a unique solution*. Clearly, all 'interesting' models of the axioms would satisfy RSP. (We have proved it for the model $\mathcal{A}$(FS) of an extension FS of FS_Δ, in Section 4.2.7.)

3.3.6. PROPOSITION. $FS_\Delta - DE4 + RSP \vdash \Delta(x+y) = \Delta x + \Delta y$ (DE4)

PROOF. Consider the following guarded systems of recursion equations, where i does not occur in x,y:

$$E_1 \begin{cases} p = \tau p_1 + \tau p_2 \\ p_1 = x + ip_1 \\ p_2 = y + ip_2 \end{cases} \qquad E_2 \begin{cases} q = \tau q_1 + \tau q_2 \\ q_1 = x + iq_2 \\ q_2 = y + iq_1 \end{cases}$$

$$E_3 \begin{cases} r = \tau r_1 + \tau r_2 \\ r_1 = x + ir_1 + ir_2 \\ r_2 = y + ir_1 + ir_2 \end{cases}$$

Then the solution vector (p, r'_1, r'_2) satisfies E_3, where $r'_1 = x + ip_1 + ip_2$ and $r'_2 = y + ip_1 + ip_2$:

$$p = \tau(x+ip_1) + \tau(y+ip_2) \underset{R1}{=} \tau(x+ip_1+ip_2) + \tau(y+ip_1+ip_2) = \tau r'_1 + \tau r'_2$$

$$r'_1 = x + ip_1 + ip_2 = x + i(x+ip_1) + i(y+ip_2) \underset{R1}{=}$$
$$x + i(x+ip_1+ip_2) + i(y+ip_1+ip_2) = x + ir'_1 + ir'_2$$

$$r'_2 = y + ip_1 + ip_2 = x + i(x+ip_1) + i(y+ip_2) =$$
$$y + i(x+ip_1+ip_2) + i(y+ip_1+ip_2) = y + ir'_1 + ir'_2.$$

Likewise one calculates that (q, r''_1, r''_2) satisfies E_3, where $r''_1 = x + iq_1 + iq_2$ and $r''_2 = y + iq_1 + iq_2$. Since E_3 is guarded, RSP applies and yields equality of these two solution vectors for E_3. In particular, p = q; and hence

$$\tau_{\{i\}}(p) = \tau_{\{i\}}(q) \qquad (1)$$

Further, using E_1 we have

$$\tau_{\{i\}}(p) = \tau_{\{i\}}(\tau p_1 + \tau p_2) = \tau\cdot\tau_{\{i\}}(p_1) + \tau\cdot\tau_{\{i\}}(p_2) \underset{DE_1}{=}$$
$$\tau\cdot\Delta x + \tau\cdot\Delta y = \Delta x + \Delta y \qquad (2)$$

Here DE_1 (a consequence of DE^∞) is used as follows:

$$\frac{p_1 = ip_1 + x}{\tau_{\{i\}}(p_1) = \Delta(\tau_{\{i\}}(x)) = \Delta(x)} \quad DE_1$$

(Note that $\tau_{\{i\}}(x) = x$ by the assumption that i does not occur in x. In fact, a formal justification of this equation would necessitate a calculus to deal with alphabets and axioms and rules thereabout; these can be found in BAETEN, BERGSTRA & KLOP [1]. Here, we will treat this matter in an informal way.)

Similarly, E_2 gives

$$\tau_{\{i\}}(q) = \tau\cdot\tau_{\{i\}}(q_1) + \tau\cdot\tau_{\{i\}}(q_2) \underset{DE_2}{=}$$
$$\tau\cdot\Delta(x+y) + \tau\cdot\Delta(x+y) = \tau\Delta(x+y) = \Delta(x+y). \qquad (3)$$

Here we used DE_2:

$$\frac{q_1 = iq_2 + x, \quad q_2 = iq_1 + y}{\tau_{\{i\}}(q_1) = \tau_{\{i\}}(q_2) = \Delta(\tau_{\{i\}}(x+y)) = \Delta(x+y)} \quad DE_2$$

Now by (1)-(3):

$$\Delta(x+y) = \Delta x + \Delta y. \quad \square$$

3.3.7. REMARK. The following is a pictorial representation of the essence of the proof in Proposition 3.3.6 (see Figure 8):

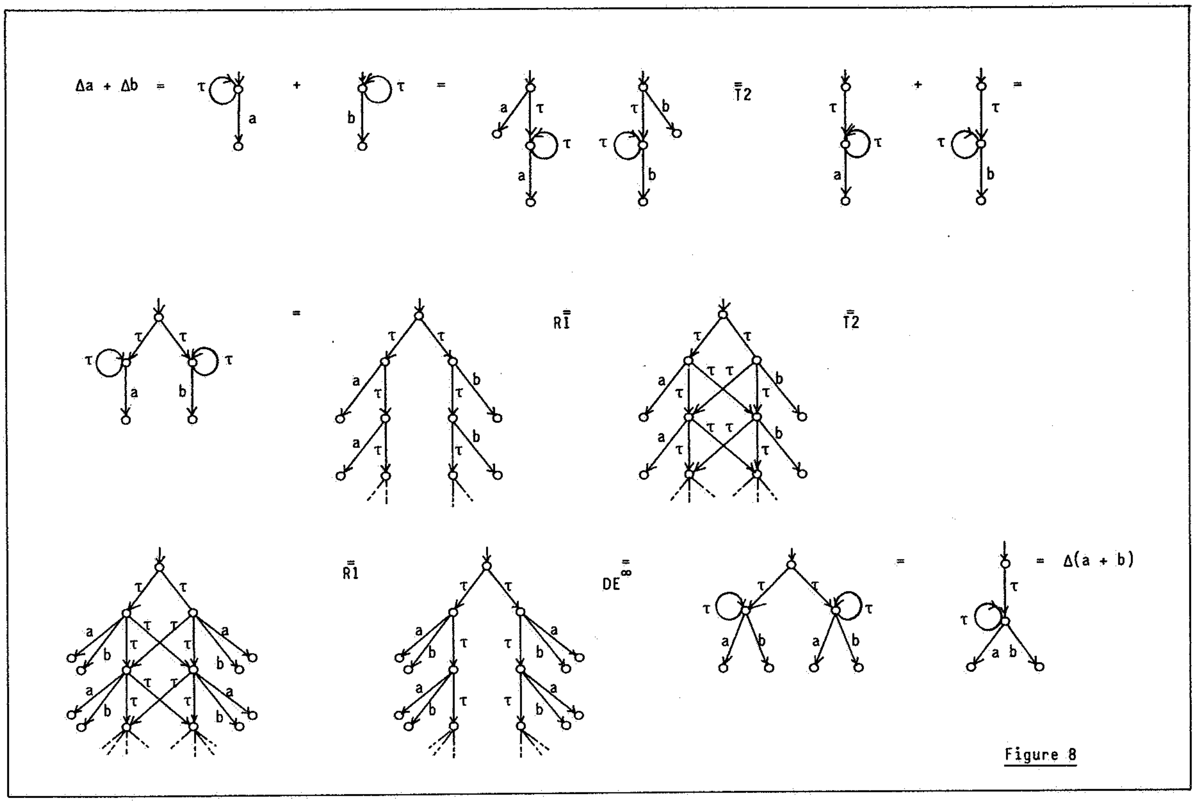


Figure 8

3.3.8. PROPOSITION. *The following are derivable in* FS_Δ*:*

(i) $\Delta x = \Delta x + \Delta\delta$

(ii) $a\Delta(x+y) = a\Delta x + a\Delta y$

(iii) $a\Delta(x+y) = a\Delta x + a\Delta y + a\Delta\delta$

PROOF. Simple. □

It is important to realize that the present extension of the kernel system BS_Δ, is inconsistent with the extension BS (= $BS_\Delta + \{\Delta = \tau\}$):

3.3.9. PROPOSITION. $BS + FS_\Delta$ *is trace inconsistent.*

PROOF. $BS + FS_\Delta \vdash a(x+y) = a\tau(x+y) = a\Delta(x+y) = a\Delta x + a\Delta y = a\tau x + a\tau y = ax + ay$, and for e.g. $x = b$, $y = \delta$ this yields $ab = ab + a\delta$, a trace inconsistency in the sense of Section 1.4. □

3.4. *A model for failure semantics with explicit divergence.*

As before, we want to construct a model for the axioms in Table 10. In fact, a model is constructed in Section 4.2 below, so we do not need to be worried about the consistency of the axioms in Table 10. However, this model $\mathcal{A}$(FS) cannot be called a model for explicit divergence, as it satisfies $\Delta\tau = \tau$ with the consequence that in this model we abstract from many divergences. Here a 'divergence' is defined as follows.

3.4.1. DEFINITION. A process graph is called *divergent* if it is possible to perform an infinite sequence of τ-steps, starting from the root. A node in a process graph is divergent if the corresponding subgraph is.

We will discuss some of the problems in the construction of a model for FS_Δ, the axiom system in Table 10; further we will present a conjectured model for FS_Δ which respects divergence. First let us be more precise about the phrase "a model with explicit divergence" or "a model which respects divergence".

3.4.2. DEFINITION. Let t,s be expressions which are δ-normalised. Then a model $\mathcal{A}$ *respects divergence* if $\mathcal{A} \models t = s$ implies that both t,s contain an occurrence of Δ or both do not.

The most straightforward attempt to construct a model for FS_Δ which respects divergence, is to consider all process graphs $g \in \mathcal{G}_{\aleph_1}$ and define the failure set $\mathcal{F}[\![g]\!]$ as in the following Table.

3.4.3. DEFINITION. Let $g \in \mathcal{G}_{\aleph_1}$. The failure set $\mathcal{F}[\![g]\!]$ is defined as the least set satisfying the clauses in Table 11. For all $\sigma \in A^*$:

TABLE 11. Failure set for processes with explicit divergence

(i)	$g \xrightarrow{\sigma} h$, h not divergent $\Rightarrow$ $(\sigma, \overline{I(h)}) \in \mathcal{F}[\![g]\!]$
(ii)	$g \xrightarrow{\sigma} \mathbf{0}$ $\Rightarrow$ $\sigma\varepsilon \in \mathcal{F}[\![g]\!]$
(iii)	$(\sigma, X \cup Y) \in \mathcal{F}[\![g]\!]$ $\Rightarrow$ $(\sigma, X) \in \mathcal{F}[\![g]\!]$
(iv)	g unstable $\Rightarrow$ $\tau \in \mathcal{F}[\![g]\!]$
(v)	$g \xrightarrow{\sigma} h$, h divergent $\Rightarrow$ $(\sigma, \Delta) \in \mathcal{F}[\![g]\!]$

Now define for $g,h \in \mathcal{G}_{\aleph_1}$: $g \equiv_{\mathcal{F}} h$ iff $\mathcal{F}[\![g]\!] = \mathcal{F}[\![h]\!]$. (In words, g,h are failure equivalent.) The problem, pointed out to us by R. van Glabbeek, is that this notion of failure equivalence is not a congruence on $\mathcal{G}_{\aleph_1}$. The difficulty is in the abstraction operator τ_I. Namely, let g,h be as in Figure 9 (in an ad hoc notation:

$$g = \sum_{n\geqslant 1} a^n \text{ and } h = \sum_{n\geqslant 1} a^n + a^{\omega}.)$$

Then $g \equiv_{\mathcal{F}} h$. However,

$$\tau_{\{a\}}(g) \not\equiv_{\mathcal{F}} \tau_{\{a\}}(h)$$

since $\tau_{\{a\}}(h)$ contains (λ,Δ) and $\tau_{\{a\}}(g)$ does not.

Of course, the problem is due to the fact that the definition of the failure set in Table 11 works on a local scale when process graphs are concerned without divergences - hence g,h are failure equivalent. But when divergences are around, the definition of the failure set turns out to possess infinite foresight: it can discern divergences. Therefore abstraction does not work.

A remedy seems to be to add a clause (vi) to Table 11: "If $\rho \in A^{\omega}$ is an infinite trace of g, then $\rho \in \mathcal{F}[\![g]\!]$". The effect is that now g,h of Figure 9 are no longer failure equivalent.

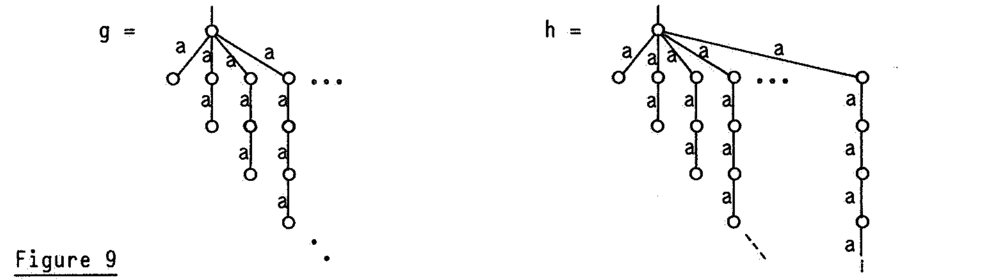


Figure 9

However, with this revised failure equivalence there is a new problem. (Also this remark is due to R. van Glabbeek.) Let $\bar{g},\bar{h}$ be the equivalence classes of g,h from Figure 9. Then both $\bar{g}$ and $\bar{h}$ solve the equation X = aX + a; it is easy to check that $g \equiv_{\mathcal{F}} a.g + a$ and likewise for h. Since $\bar{g} \neq \bar{h}$ this means that the purported model $\mathcal{G}_{\aleph_1}/\equiv_{\mathcal{F}}$ would not satisfy RSP, which in itself is unsatisfactory.

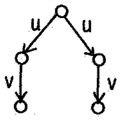
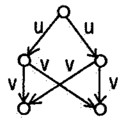
We will now discuss a construction which, we conjecture, does give a satisfactory model. (The discussion is tentative and further investigation is required.)

3.4.4. DEFINITION. Let $g \in \mathcal{G}_{\aleph_1}$.

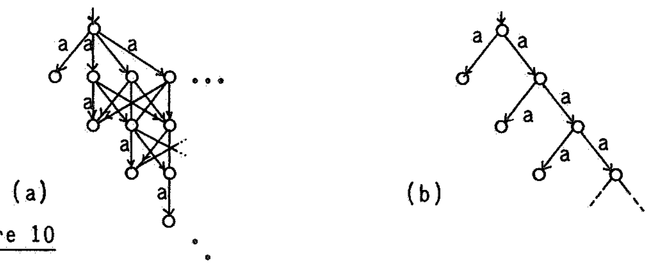
(i) g is *weakly divergent* if there are, starting from the root, arbitrarily long τ-paths.

(ii) g is *weakly divergent after* σ if there are, starting from the root, arbitrarily long paths π with $val(\pi) = \sigma$. Notation: $g \xrightarrow{\sigma} \Delta$.

Now define a variant of $\mathcal{F}[\![\ ]\!]$ as in Table 11, by replacing clause (v) by: (v)' $g \xrightarrow{\sigma} \Delta \implies (\sigma,\Delta) \in \mathcal{F}[\![g]\!]$. We write for the resulting failure equivalence: $\equiv^{w}_{\mathcal{F}}$. *)

Interestingly, the difference between divergence and weak divergence seems to fade away in failure semantics, since every weakly divergent graph g is failure equivalent (in the sense of $\equiv^{w}_{\mathcal{F}}$) to a divergent graph g'. This can be seen by saturating g with 'crosses' (see [8]): every part  $(u,v \in A_\tau)$ may be replaced by  without altering the failure set.

Call a graph 'cross-saturated' if adding a cross yields a bisimilar graph. So the cross-saturation of g as in Figure 9 is g' as in Figure 10(a) and this graph is in fact bisimilar to g" in Figure 10(b).



(a)

(b)

Figure 10

) Definition 3.4.4(ii) was suggested to us by R. van Glabbeek who noted that a simpler definition of $\equiv^{w}_{\mathcal{F}}$, obtained by replacing 'divergent' by 'weakly divergent' in Table 11, does not work. For, let g, h* be graphs as g,h in Figure 9 but with all initial steps relabeled into 'b'. Then we would have (in the simpler definition of $\equiv^{w}_{\mathcal{F}}$) that g* and h* are failure equivalent. However, $\tau_{\{a\}}(g^*) = b$ while $\tau_{\{a\}}(h^*) = b + b\Delta$; hence (b,Δ) would be in the failure set of h* but not in that of g*. So:

$$\tau_{\{a\}}(g^*) \not\equiv^{w}_{\mathcal{F}} \tau_{\{a\}}(h^*);$$

in other words, abstraction is not well-defined.

3.4.5. <u>CONJECTURE</u>. *$\mathcal{G}_{\aleph_1}/\equiv^{w}_{\mathcal{F}}$ is a model for* FS_Δ *with explicit divergence, and satisfying RSP.*

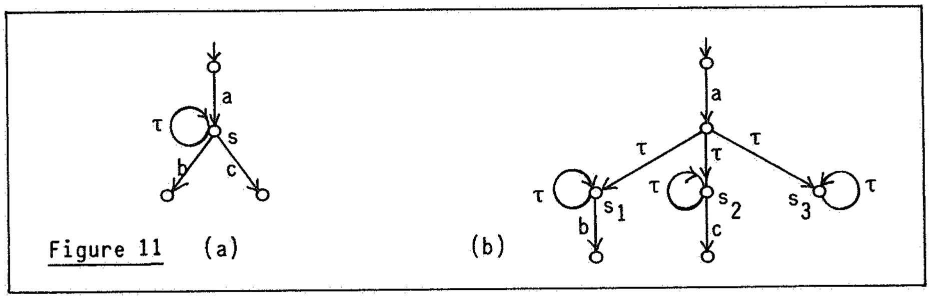
Note that, since every g is failure equivalent (in the sense of $\equiv^{w}_{\mathcal{F}}$) to a cross-saturated g', an equivalent construction is obtained by restricting $\mathcal{G}_{\aleph_1}$ to $\mathcal{G}^{x}_{\aleph_1}$, the domain of cross-saturated process graphs, and now dividing out $\equiv_{\mathcal{F}}$ as given by Table 11. In other words:

$$\mathcal{G}_{\aleph_1}/\equiv^{w}_{\mathcal{F}} \text{ and } \mathcal{G}^{x}_{\aleph_1}/\equiv_{\mathcal{F}} \text{ are isomorphic.}$$

It would be interesting to investigate the relationship between the notion of 'cross-saturated' process graph and suitable notions of 'closed' process graphs.

It is also interesting to note that the model $\mathcal{A}$(FS), constructed below in section 4.2, which satisfies $\Delta\tau = \tau$, does not need a restriction of the underlying process graph domain.

3.4.6. <u>REMARK</u>. The definition of $\mathcal{F}[\![\ \ ]\!]$ in Table 11 states that in a divergence node only the trace σ and the information 'divergence' (Δ) is given and no information about impossible next steps, as in clause (i). This is not an arbitrary choice but a rather straightforward consequence of previous considerations, notably the linearity of Δ. Namely, consider the example $a\Delta(b + c)$ or its graph in Figure 11(a):



<u>Figure 11</u> (a) (b)

As the definition of failure set stands, the contribution of node s in g is (a, Δ). Indeed the additional information in, say, $(a, \overline{\{b,c\}}, \Delta)$ or rather

$$\{(a, X, \Delta) \mid X \subseteq \overline{\{b,c\}}\}$$

would be superfluous, since by linearity of Δ the process $a\Delta(b+c)$ equals $a(\Delta b + \Delta c + \Delta\delta)$, with graph h in Figure11b. Now nodes s_1, s_2, s_3 in h would yield

$$(a, \overline{\{b\}}, \Delta), \quad (a, \overline{\{c\}}, \Delta), \quad (a, \overline{\emptyset}, \Delta)$$

and hence the information about impossible initial steps is obliterated since

$$\{(a, X, \Delta) \mid X \subseteq \overline{\{b,c\}}\} \subseteq \{(a, Y, \Delta) \mid Y \subseteq \overline{\emptyset}\}.$$

3.5. *Process graph transformations.*

In BERGSTRA, KLOP & OLDEROG [8] we have employed certain *process graph transformations* in order to get a completeness result. A typical process graph transformation used there, and which is also valid here, is:

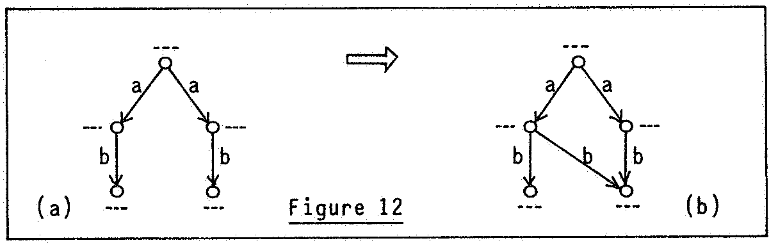


Figure 12

That is, in a configuration occurring in a process graph as in Figure 12(a), a b-step may be inserted as in Figure 12(b). Here $a,b \in A_{\delta\tau}$.
Note that two applications of this transformation just correspond to the axiom R1 in Table 7:

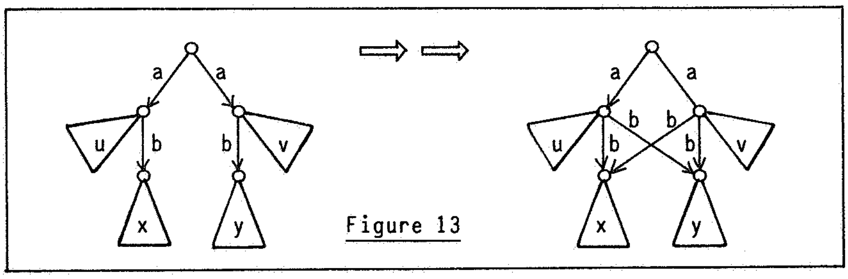


Figure 13

It is not hard to verify that this process graph transformation is sound w.r.t. $\mathcal{F}[\![\ ]\!]$, also in the present situation where a,b may be τ.

It is not required in the process graph transformation as shown that all five nodes involved are distinct; but it is required that the graph is 'history-unambiguous', i.e. a node may have only one history $\sigma \in A^*$ (note that the history σ contains only proper steps, no τ-steps). E.g. the following transformation is not sound:

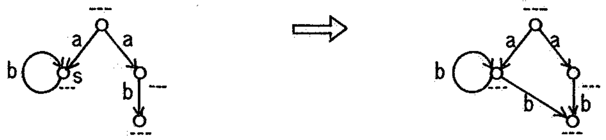


Figure 14

The transformation is not sound, as node s has histories ab*. However, if b = τ, s *is* history-unambiguous and we have an interesting sound process graph transformation which we will call the "τ-*jump*":

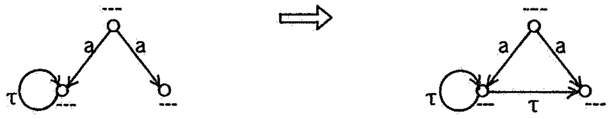


Figure 15

It is derived from the transformation explained above, by first inserting a τ-step which later can be removed:

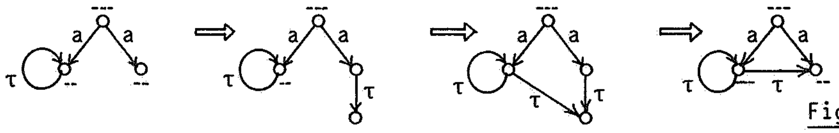


Figure 16

The τ-jump gives an easy "explanation" of the linearity of Δ:

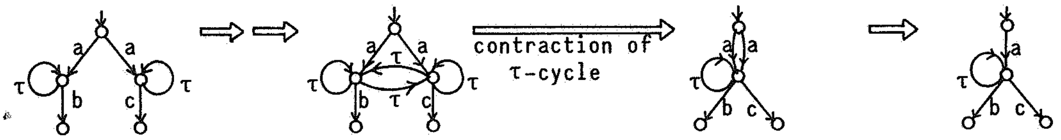
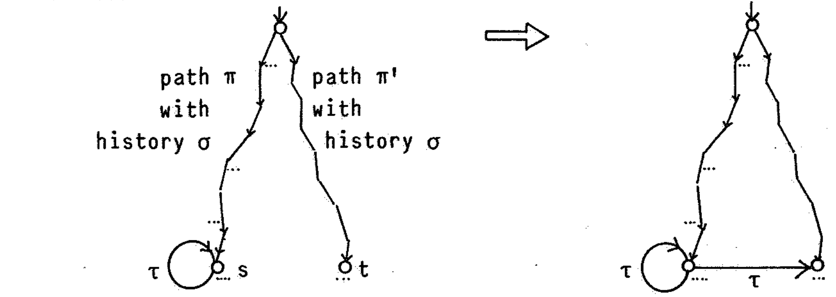


Figure 17

Note that a generalisation of the τ-jump is as follows:

Figure 18



Here nodes s,t have the same history $\sigma \in A^*$. The procedure to obtain this τ-jump is to use interpolating deterministic τ-steps to make π, π' contain the same steps including τ-steps, and then to use the 'cross' transformation in Figure 13.

An interesting consequence is that the τ-jump also "explains" (or illustrates) the identity $\Delta x = x + \Delta\delta$ (proved in Proposition 3.3.2(i)):

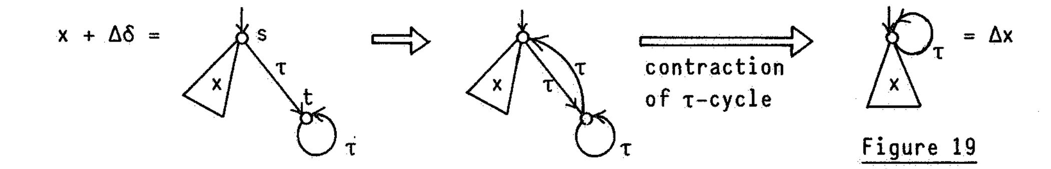


Figure 19

(Here the τ-jump is justified since s,t both have history λ.)

4. FAILURE SEMANTICS WITH FAIR ABSTRACTION OF UNSTABLE DIVERGENCE: FS

In this section we come to the main observations of this paper. In Section 2 it was shown that bisimulation semantics admits abstraction from all (periodical) divergences, formalised by the rules DE_k ($k \geqslant 1$) and $\Delta = \tau$. In failure semantics putting $\Delta = \tau$ leads at once to an inconsistency (Proposition 3.3.9); but it is possible to abstract from a special type of divergences, which we will call 'unstable' divergences, and which is formalised by the rule DE^∞ and the equation

$$\Delta\tau = \tau.$$

4.1. *An axiom and derived equations for abstraction of unstable divergence.*

Above, a process was called unstable if it admits an initial τ-step. Consider such an unstable process p, and make the process divergent by appending a τ-loop at the root, i.e. form Δp. Then Δp is a diverging process and we will say that it is an *unstable divergence* since p is. In Proposition 3.2.6 it was remarked that p is unstable iff $p = \tau p$; so $\Delta p = \Delta\tau p$. That is, unstable divergence 'is' just $\Delta\tau$.

4.1.1. PROPOSITION. *The axiom system* FS_Δ *in Table 10 is inconsistent with* $\Delta = \tau$, *but is consistent with* $\Delta\tau = \tau$.

PROOF. The first part of the proposition is proved in Proposition 3.3.9; in Section 4.2 a model $\mathcal{A}$(FS) for fair abstraction from unstable divergence is given. □

We will refer to the equation $\Delta\tau = \tau$ as '*fair abstraction from unstable divergence*'. The axiom system $FS_\Delta + \{\Delta\tau = \tau\}$ will be denoted by FS.

4.1.2. *Equations derived from* FS.

In FS we can derive the following list of equations (see Table 12). For the sake of completeness we have included in Table 12 some derived equations mentioned earlier, viz. some that are already derivable without the axiom $\Delta\tau = \tau$. Also several of these equations are formulated (in the right column of Table 12) in equivalent versions using ε (e.g. $\Delta = \varepsilon + \Delta\delta$ is equivalent with $\Delta x = x + \Delta\delta$, as can be seen by evaluating $\Delta x = (\varepsilon + \Delta\delta)x$). Moreover, ε enables us to treat Δ as a constant, so that e.g. the equation $\tau x = \tau x + \Delta x$ can be rendered as $\tau = \tau + \Delta$.

TABLE 12. Derived equations in failure semantics with fair abstraction

without Δ		with ε	
$x+\tau y = \tau(x+\tau y)$	1		
$\tau(ax+ay) = \tau ax+ay$	2		
$ax+ay = a(\tau x+\tau y)$	3		
$\tau x = \tau x+x$	4	$\tau = \tau+\varepsilon$	4'
$\tau(x+y) = \tau(x+y)+x$	5		
$a(x+\tau y) = a(x+\tau y)+ay$	6		
$a(x+\tau y) = a(x+y)+ay$	7	$a(\tau+x) = a(\varepsilon+x)+a$	7'
without $\Delta\tau = \tau$			
$\Delta x+y = \tau(\Delta x+y)$	8		
$\Delta x = \Delta x+\Delta\delta$	9	$\Delta = \Delta+\Delta\delta$	9'
$a\Delta x = a\Delta x+a\Delta\delta$	10		
$a\Delta(x+y) = a\Delta x+a\Delta y$	11		
$a\Delta(x+y) = a\Delta x+a\Delta y+a\Delta\delta$	12		
with $\Delta\tau = \tau$			
$\tau x = \tau x+\Delta x$	13	$\tau = \tau+\Delta$	13'
$\tau x = \tau x+\Delta\delta$	14	$\tau = \tau+\Delta\delta$	14'
$\Delta x = x+\Delta\delta$	15	$\Delta = \varepsilon+\Delta\delta$	15'
$\Delta(x+\tau y) = x+\tau y$	16		
$\Delta x+\tau y = x+\tau y$	17		
$a\Delta x+ay = a(x+\tau y)$	18		
$ax = ax+a\Delta\delta$	19	$a = a+a\Delta\delta$	19'
$ax = ax+a\Delta x$	20	$a = a+a\Delta$	20'
$a(\tau+x) = a+a\Delta x$	21		
$\tau+x = \tau+\Delta x$	22		
$\Delta(x+y) = x+\Delta y$	23		
		$\Delta\Delta = \Delta$	24

PROOF. Equations 1-12 have been proved in Propositions 3.2.9 and 3.3.8. As to the remaining identities, they can easily be proved from the axioms together with 1-12 and the crucial identity 15. We will prove the equivalent form 15', $\Delta = \varepsilon + \Delta\delta$:

$$\begin{aligned}
&\Delta = \\
&\Delta\varepsilon = \\
&\Delta(\varepsilon + \delta) = \\
&\Delta\varepsilon + \Delta\delta = \\
&\Delta\varepsilon + \tau\Delta\delta = \\
&\Delta\varepsilon + \Delta\tau\Delta\delta = \\
&\Delta(\varepsilon + \tau\Delta\delta) = \\
&\Delta(\tau(\varepsilon + \Delta\delta) + \tau\Delta\delta\ ) = \\
&\Delta\tau(\varepsilon + \Delta\delta) + \Delta\tau\Delta\delta = \\
&\tau(\varepsilon + \Delta\delta) + \tau\Delta\delta = \\
&\varepsilon + \tau\Delta\delta = \\
&\varepsilon + \Delta\delta
\end{aligned}$$

Some examples of proofs for some other identities may suffice:

13. $\tau x = \Delta\tau x = \Delta(\tau x + x) = \Delta\tau x + \Delta x = \tau x + \Delta x$.

14. $\tau x = \tau x + \Delta x = \tau x + x + \Delta\delta = \tau x + \Delta\delta$.

15. $\Delta x + \tau y = x + \tau y + \Delta\delta = x + \tau y$.

24. $\Delta\Delta = \Delta\tau\Delta = \tau\Delta = \Delta$. Alternatively: $\Delta\Delta = \Delta + \Delta\delta = \Delta$. □

Note that identity 15, which is DE3 in Table 10, is already proved in Proposition 3.3.2(i), in the context of FS_Δ. The merit of the present proof of 15 (or 15') is that it does not use the infinitary rule DE^∞; instead we have used above $\Delta\tau = \tau$. We do not know whether there is a proof of DE3 in FS_Δ, without DE^∞.

4.1.3. REMARK. We recall that: p is unstable $\Longleftrightarrow$ $p = \tau p$. Likewise, p is divergent $\Longleftrightarrow$ $p = \Delta p$. (The implication $\Longrightarrow$ follows from DE^∞.) Now one easily checks that $\Delta\tau = \tau$ is equivalent to the coincidence of these properties, of being unstable and being divergent:

$$\Delta\tau = \tau \iff (\forall p \quad p \text{ is divergent iff } p \text{ is unstable}).$$

4.1.4. REMARK. From DE^{∞} (in Table 9) and $\Delta\tau = \tau$ one derives easily the following rule $KFAR^{-}$, describing "KFAR for failure semantics with fair abstraction of unstable divergence":

$$KFAR^{-} \quad \frac{x_n = i_n \cdot x_{n+1} + \tau y_n, \quad n \in \mathbb{N}, \quad i_n \in I}{\tau_I(x_0) = \tau_I(\sum_n \tau y_n)}$$

(For KFAR, see the introduction of Section 2.)

An inspection of the details of the algebraical verification of the Alternating Bit Protocol in [5], which was done in the setting of bisimulation semantics using KFAR, learns that this verification also could have taken place in failure semantics using $KFAR^{-}$.

4.2. *A model for failure semantics with abstraction of unstable divergence.*

We will now define a homomorphic image of the model $\mathcal{A}(FS_{\Delta})$ in Section 3.4 for failure semantics with explicit divergence; the new model will be obtained by abstraction from unstable divergence in a sense made precise below. (We will not prove here that the new model is in fact a homomorphic image of $\mathcal{A}(FS_{\Delta})$.) As in Section 3.4 the point of departure is the domain of countably branching process graphs; the notion of failure equivalence between such process graphs which we are going to define arises by adapting clause (v) in Table 10: for $g \in \mathcal{G}$, $\mathcal{F}^*\llbracket g \rrbracket$ is defined as the least set such that for all $\sigma \in A^*$ the clauses in Table 13 hold.

TABLE 13. Failure set for processes with abstraction of unstable divergence

(i)	$g \xrightarrow{\sigma} h$, h not divergent $\Rightarrow$ $(\sigma, \overline{I(h)}) \in \mathcal{F}^*\llbracket g \rrbracket$
(ii)	$g \xrightarrow{\sigma} 0$ $\Rightarrow$ $\sigma\varepsilon \in \mathcal{F}^*\llbracket g \rrbracket$
(iii)	$(\sigma, X \cup Y) \in \mathcal{F}^*\llbracket g \rrbracket$ $\Rightarrow$ $(\sigma, X) \in \mathcal{F}^*\llbracket g \rrbracket$
(iv)	g unstable $\Rightarrow$ $\tau \in \mathcal{F}^*\llbracket g \rrbracket$
(v)*	$g \xrightarrow{\sigma} h$, h divergent, $\forall X \subseteq A_{\varepsilon}$ $(\sigma, X) \notin \mathcal{F}^*\llbracket g \rrbracket$ $\Rightarrow$ $(\sigma, \Delta) \in \mathcal{F}^*\llbracket g \rrbracket$

The rather complicated clause (v)* (which will be eliminated soon) says that a (σ, Δ) obtained from a diverging node s with history σ, may be erased from $\mathcal{F}^*[g]$ whenever there is a 'proper' failure pair (σ, X) around, i.e. whenever g contains a non-diverging node s' with the same history σ.

4.2.1. <u>EXAMPLES</u>. (i) In g as in Figure 20(a), the contribution of diverging node s, namely (a, Δ), may be erased as there is a non-diverging node s' with the same history. Thus

$$\mathcal{F}^*[\![g]\!] = \{\tau, (\lambda, \overline{\{a\}}), (a, \overline{\{b\}}), (ab, \overline{\{\varepsilon\}}), ab\varepsilon\}^c$$

So if h,k are as in Figure 20(b),(c), then

$$\mathcal{F}^*[\![g]\!] = \mathcal{F}^*[\![h]\!] = \mathcal{F}^*[\![k]\!].$$

In the algebra we have

$$a\Delta\tau b = a\tau b = ab.$$

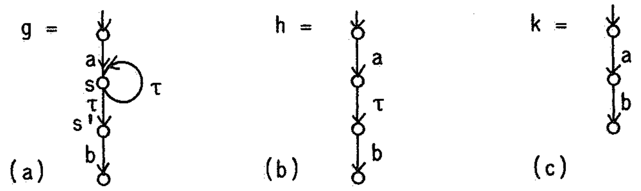


Figure 20

(ii) Similarly, if g is the graph in Figure 21(a), then $\mathcal{F}^*[\![g]\!]$ does not contain (a, Δ) since s' is non-diverging and has history a. It is easy to check that $\mathcal{F}^*[\![g]\!]$ equals the failure set of the graphs in Figure 21(b)-(e). Algebraically:

$$a\Delta(b+\tau c) + ac = a\Delta\tau(b+\tau c) + ac = a\tau(b+\tau c) + ac = a(b+\tau c) + ac.$$

Note that in both examples (i),(ii) we find that $\Delta\tau = \tau$.

As always, we define

$$g \equiv_{\mathcal{F}^*} h \iff \mathcal{F}^*[\![g]\!] = \mathcal{F}^*[\![h]\!].$$

It turns out that there is a very natural definition of $\equiv_{\mathcal{F}^*}$, via the failure set $\mathcal{F}[\![\ ]\!]$ as defined in the following Table 14. The remarkable property of this definition is that it does not mention the divergence property of nodes at all.

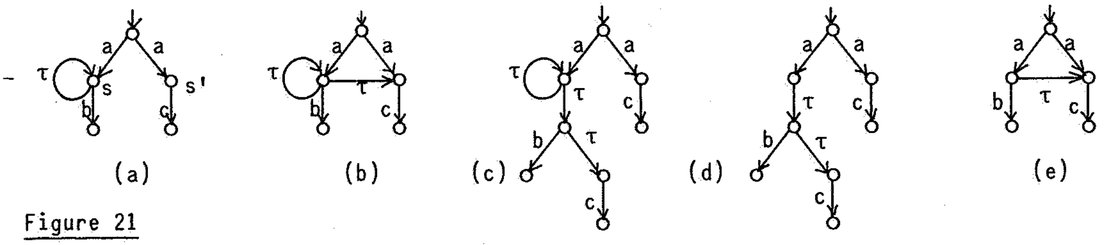


Figure 21

4.2.2. DEFINITION. (i) $\mathcal{F}\llbracket g\rrbracket$ is defined as the least set such that for all $\sigma \in A^*$ the following clauses hold:

TABLE 14. Failure set for processes with abstraction of unstable divergence

(i)	$g \xrightarrow{\sigma} h$, h stable $\Rightarrow$ $(\sigma, \overline{I(h)}) \in \mathcal{F}\llbracket g\rrbracket$
(ii)	$g \xrightarrow{\sigma} \emptyset$ $\Rightarrow$ $\sigma\varepsilon \in \mathcal{F}\llbracket g\rrbracket$
(iii)	$(\sigma, X \cup Y) \in \mathcal{F}\llbracket g\rrbracket$ $\Rightarrow$ $(\sigma, X) \in \mathcal{F}\llbracket g\rrbracket$
(iv)	g unstable $\Rightarrow$ $\tau \in \mathcal{F}\llbracket g\rrbracket$
(v)*	$g \xrightarrow{\sigma} h$ $\Rightarrow$ $\sigma \in \mathcal{F}\llbracket g\rrbracket$

(ii) $g \equiv_{\mathcal{F}} h \Leftrightarrow \mathcal{F}\llbracket g\rrbracket = \mathcal{F}\llbracket h\rrbracket$.

4.2.3. PROPOSITION. *Let $\equiv_{\mathcal{F}^*}$ be defined as in Table 13 and $\equiv_{\mathcal{F}}$ as in Table 14. Then $\equiv_{\mathcal{F}^*}$ and $\equiv_{\mathcal{F}}$ coincide.*

PROOF. We will transform the definition of $\equiv_{\mathcal{F}^*}$ via Table 13 in a number of steps into the definition of $\equiv_{\mathcal{F}}$ via Table 14.

(1) The first transformation of the definition in Table 13 consists of remarking that instead of *erasing* (σ, Δ) whenever a (σ, X) is present, one can without altering the notion of $\equiv_{\mathcal{F}^*}$, *add* (σ, Δ) whenever a $(\sigma, X)$is present. Thus we obtain the definition

(i)-(iv) as in Table 13

(v) $g \xrightarrow{\sigma} h$, h divergent $\Rightarrow (\sigma, \Delta) \in \mathcal{F}^*[\![g]\!]$

(vi) $(\sigma, X) \in \mathcal{F}^*[\![g]\!] \Rightarrow (\sigma, \Delta) \in \mathcal{F}^*[\![g]\!]$

(2) This definition is logically equivalent to the definition:

(i)-(iv) as in Table 13

(v)' $g \xrightarrow{\sigma} h \Rightarrow (\sigma, \Delta) \in \mathcal{F}^*[\![g]\!]$

(3) Instead of adding (σ, Δ) we obtain the same information if σ is added:

(i)-(iv) as in Table 13

(v)" $g \xrightarrow{\sigma} h \Rightarrow \sigma \in \mathcal{F}^*[\![g]\!]$

(4) The next transformation is less trivial and consists in replacing the proviso "h is not divergent" of clause (i) in Table 13 by "h is not unstable" (or "h is stable"). Note that this replacement is in accordance with Remark 4.1.3. This last transformation gives us Table 14. Not only does this transformation keep $\equiv_{\mathcal{F}}$ the same, it even keeps $\mathcal{F}[\![\ ]\!]$ the same.

To see this, we argue as follows. Let $\mathcal{F}^*[\![g]\!]$ be the failure set defined as in (3), and $\mathcal{F}[\![g]\!]$ as defined in Table 14. The only difference between $\mathcal{F}^*[\![g]\!]$ and $\mathcal{F}[\![g]\!]$ is that $\mathcal{F}[\![g]\!]$ contains (a priori) less (σ, X)'s than $\mathcal{F}^*[\![g]\!]$; namely, nondiverging but unstable nodes give no contribution in $\mathcal{F}[\![g]\!]$. However this does not matter: if s is a nondiverging unstable node, there must be a stable node t below s, reachable from s by τ-steps. Now it is evident that the contribution at s is contained by that at t. □

4.2.4. <u>PROPOSITION</u>. $\equiv_{\mathcal{F}}$ *is a congruence w.r.t. the operations* $+, \cdot, \tau_I$ *in the process graph domain* $\mathcal{G}_{\aleph_1}$.

<u>PROOF</u>. As a preliminary notion, define for a process graph g:

$$FP(g) = \{(\sigma, X) \mid (\sigma, X) \in \mathcal{F}[\![g]\!]\}$$

$$TR(g) = \mathcal{F}[\![g]\!] - FP(g).$$

So FP(g) is the part of the failure semantics of g consisting of failure pairs and TR(g) is the 'trace part'; we have:

$$\mathcal{F}[\![g]\!] = \mathcal{F}[\![g']\!] \iff \mathrm{FP}(g) = \mathrm{FP}(g') \;\&\; \mathrm{TR}(g) = \mathrm{TR}(g').$$

First we consider the easiest case: τ_I.
Let $g \equiv_{\mathcal{F}} g'$; to prove $\tau_I(g) \equiv_{\mathcal{F}} \tau_I(g')$. Now

$$\mathrm{TR}(g) = (\{\tau\}) \cup \{\sigma \mid \sigma \in A^*,\ \sigma \in \mathcal{F}[\![g]\!]\} \cup \{\sigma\varepsilon \mid \sigma \in A^*,\ \sigma\varepsilon \in \mathcal{F}[\![g]\!]\}.$$

Here σ is the 'instability indicator', signaling the possible presence of an initial τ-step.

(1) To prove $\mathrm{TR}(\tau_I(g)) = \mathrm{TR}(\tau_I(g'))$.

Case 1: if g is unstable, so is g'; and hence both $\tau_I(g)$, $\tau_I(g')$ are unstable.

Case 2: g,g' both stable. Then, it is easy to see that abstraction by τ_I leads to an initial τ-step for both of $\tau_I(g)$, $\tau_I(g')$, or for none of them.

So we have proved now that

$$\tau \in \mathrm{TR}(\tau_I(g)) \iff \tau \in \mathrm{TR}(\tau_I(g')).$$

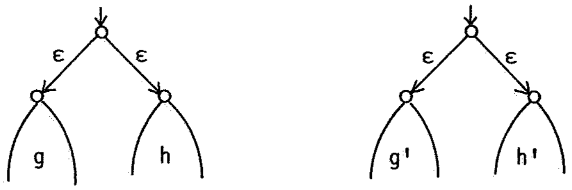
Proving that $\mathrm{TR}(\tau_I(g))$ and $\mathrm{TR}(\tau_I(g'))$ also coincide w.r.t. their "σ-part" and "σε-part" is trivial.

(2) Further, to prove $\mathrm{FP}(\tau_I(g)) = \mathrm{FP}(\tau_I(g'))$.
Let $(\sigma, X) \in \mathrm{FP}(\tau_I(g))$, and moreover let X be maximal so (for σ fixed). Then this (σ, X) was obtained from a stable node s in $\tau_I(g)$. The corresponding node s in g (which is there since $\tau_I(g)$ is merely a relabeling of g) hence has outgoing steps I(s) disjoint from I. By maximality of X, in fact $X = \overline{I(s)} = A_\varepsilon - I(s)$. So $I \subseteq X$. It follows that node s in g is also stable and yields the same (σ, X) as a contribution to the failure set $\mathcal{F}[\![g]\!]$. Since $\mathcal{F}[\![g]\!] = \mathcal{F}[\![g']\!]$, $(\sigma, X) \in \mathcal{F}[\![g']\!]$, obtained at stable node $t \in g'$. Now since $I \subseteq X$, we have $I \subseteq I(t) = \emptyset$ and therefore the stability of node t is preserved in $\tau_I(g')$. Therefore t in $\tau_I(g')$ also contributes (σ, X). So we have proved $(\sigma, X) \in \mathrm{FP}(\sigma_I(g'))$, and, by symmetry, $\mathrm{FP}(\tau_I(g)) = \mathrm{FP}(\tau_I(g'))$.

(1) & (2) $\Longrightarrow$ $\tau_I(g) \equiv_{\mathcal{F}} \tau_I(g')$.

Secondly, we consider +. So let $g \equiv_{\mathcal{F}} g'$. To prove: $g + h \equiv_{\mathcal{F}} g' + h$. By definition, g + h and g' + h are the graphs as in Figure 22, where we have anticipated the presence of ε, considered in Section 4.2.9 below. (We leave it to the reader to make the present argument "ε-free".)

Figure 22



That TR(g + h) = TR(g' + h) is easy. To check that FP(g + h) = FP(g' + h), we must look at derivations $g+h \xrightarrow{\sigma} k$. The case that $\sigma \neq \lambda$ (the empty word in A*), is not difficult since those derivations 'lead properly into' either g or h.

It remains to check the contributions by derivations $g+h \xrightarrow{\lambda} k$, $g'+h \xrightarrow{\sigma} \ell$. Now if g or h is unstable, also g' or h is unstable, and λ-derivations give no contribution, done. Otherwise, we have g,h stable and simultaneously g',h stable. This case is again easily dealt with.

The third case, for •, is left to the scrupulous reader. □

4.2.5. NOTATION. $\mathcal{G}_{\aleph_1}/{\equiv_{\mathcal{F}}} = \mathcal{A}(\mathrm{FS})$.

Without proof we state:

4.2.6. THEOREM. $\mathcal{A}(\mathrm{FS}) \models \mathrm{FS}$. □

4.2.7. *The unique solvability of guarded recursion equations in* $\mathcal{A}$(FS).

Let $E = \{X_i = T_i(X_1,\ldots,X_n) \mid i = 1,\ldots,n\}$ be a guarded system of recursion equations, where the $T_i(\vec{X})$ are expressions built from variables $X_i,\ldots,X_n$, +,•, atoms a,b,c,... ∈ A and τ,ε,δ. 'E is guarded' means that every occurrence of an X_j in some T_i is preceded by an atom a,b,c,... . (More precisely: every occurrence X_j in a RHS T_i of E is an occurrence in a subterm of the form aS, for some S and a ∈ A.) We will be slightly more liberal and also admit e.g. a system as

$$\begin{cases} X_1 = \tau X_2 + \tau X_3 \\ X_2 = aX_1 + bX_2 + cX_3X_3 + d \\ X_3 = aX_2 + b(X_3 + d) \end{cases}$$

as a guarded system, since (by replacing X_2, X_3 in the first equation by their 'bodies') this system can easily be 'developed' to a guarded one.

We use the terminology in BAETEN, BERGSTRA & KLOP [1]: RDP (Recursive Definition Principle) is the statement that every guarded E has a solution vector;

RSP (Recursive Specification Principle) states the unicity of the solution of a guarded E.

4.2.7.1. THEOREM. $\mathcal{A}(FS) \models$ RDP & RSP.

PROOF. Let $E = \{X_i = T_i(\vec{X}) \mid i = 1,\ldots,n\}$ be guarded. We will briefly indicate the *existence* of a solution vector $\underline{X}_1,\ldots,\underline{X}_n$ (i.e. RDP):

CLAIM 1. In the model $\mathcal{A}(BS_\Delta)$ of BS_Δ, RDP holds.

For the process graph model $\mathcal{A}(BS)$ of BS the validity of RDP & RSP is proved in [1]; here we need only RDP and the adaptation of this result from BS to BS_Δ is not problematic.

CLAIM 2. The present model $\mathcal{A}(FS)$ is a homomorphic image of $\mathcal{A}(BS_\Delta)$, since FS is an extension of BS_Δ.

From Claims 1 and 2 we have RDP for $\mathcal{A}(FS)$.

Uniqueness of the solution $\underline{\vec{X}}$ of E (i.e. RSP). Let $n \in \mathbb{N}$ be given. Since E is guarded, we can "develop E" to depth n. That is, by repeated substitution of the bodies of the X_i we find a system

$$E' = \{X_i = T_i'(\vec{X}) \mid i=1,\ldots,n\}$$

such that the n-th projections $\pi_n(T_i'(\vec{X}))$ are closed terms (i.e. do not contain X_i's). Here π_n is defined as in [1]; intuitively π_n cuts off everything deeper than level n, where the level is measured w.r.t. the number of atoms a,b,c,... (not τ,ε,δ!) encountered.

Obviously, the closed terms $\pi_n(T_i'(\vec{X}))$ determine an "initial part" $\mathcal{F}[\![g_i]\!] \restriction n$ of the failure sets $\mathcal{F}[\![g_i]\!]$, where the process graph g_i is a representant of $\underline{X}_i$. Here $\mathcal{F}[\![g_i]\!] \restriction n$ is defined for $n \geq 2$ as the set of σ, $\sigma\varepsilon$, (σ, X) contained in $\mathcal{F}[\![g_i]\!]$ where length$(\sigma) \leq n-1$; for $n=1$ we have to include moreover the signal 'τ' in case g_i is unstable.

Example: Let E' be $\{X = \tau abaY + cdeYX + a,\ Y = abcY\}$, then $\pi_3(\underline{X}) = \tau aba + cde + a$, $\pi_3(\underline{Y}) = abc$. Now if g,h are process graphs representing $\underline{X},\underline{Y}$, we know $\mathcal{F}[\![g]\!] \restriction 2$ and $\mathcal{F}[\![h]\!] \restriction 2$.

Finally, we observe that

$$\mathcal{F}[\![g]\!] = \bigcup_{n\geq 1} \mathcal{F}[\![g]\!] \restriction n .$$

Hence RSP follows. □

4.2.8. REMARK. (i) In the present semantics FS we have abstraction from all unstable divergences; what remains in a process, are the "proper" divergences, characterised as follows. First define, for g a process graph:

g has a proper divergence at σ ($\in A^*$) $\Longleftrightarrow$
all σ-derivatives h of g are divergent.

(If $g \xrightarrow{\sigma} h$, h is called a σ-derivative of g.)

Now one easily proves that the characterisation of proper divergences is:

g has a proper divergence at σ $\Longleftrightarrow$
$\mathcal{F}[\![g]\!]$ does not contain a failure pair (σ,X).

E.g. $ab\Delta c$ has a proper divergence 'at' $\sigma = ab$, which reveals itself in the absence of a failure pair (ab, X) in $\mathcal{F}[\![ab\Delta c]\!]$.

Likewise

$$a(b' + b\Delta c) + a(b'' + b\Delta d) =$$
$$a(b\Delta c + b\Delta d + b') + a(b\Delta c + b\Delta d + b'') =$$
$$a(b\Delta(c+d) + b') + a(b\Delta(c+d) + b'')$$

has a proper divergence 'at' $\sigma = ab$. On the other hand, the expression

$$ab\Delta c + abde$$

does not denote a process with a proper divergence, since one of the ab-derivatives, viz. de, is not divergent. Indeed:

$$ab\Delta c + abde = a(b\Delta c + bde) = a(b\tau\Delta c + b\tau de) = a(b\tau\Delta c + b\Delta\tau de) =$$
$$ab(\Delta c + \Delta\tau de) = ab\Delta(c + \tau de) = ab\Delta\tau(c + \tau de) = ab(c + \tau de) =$$
$$ab(de + c) + abde,$$

and the last expression manifestly does not contain a divergence.

(ii) The model $\mathcal{A}$(FS) of FS defined above satisfies RSP, as shown in 4.2.7. As one may expect, a process having a proper divergence can never be the solution of a guarded system of recursion equations.

Proof sketch of this proposition: suppose p has a proper divergence at σ and is a purported solution of E, a system of guarded recursion equations. Then 'developing' E to a depth > length(σ), which is possible by the assumption of guardedness, we see (by the criterion in (i) for proper divergences) that the n-th projection of p must be divergence-free.

4.2.9. *Extension with the empty process* ε.

As before, we consider the extension with ε In the process graph domain this means that ε may occur as label; for simplicity we will not admit infinite ε-traces. (The reason is that there is a degree of freedom on how to treat infinite ε-traces. One plausible option is to arrange that $\varepsilon^{\omega} = \delta$; we prefer here to save us these choices.)

In the algebra, we have again as only extra equations

$$x\varepsilon = x$$
$$\varepsilon x = x$$
$$\tau_I(\varepsilon) = \varepsilon.$$

Just as δ,τ the new constant ε will not be an element of A, the set of proper actions a,b,c,... .

As to the semantics: we will extend the definition of $F[\![g]\!]$, given in Table 14, so that it covers the case where g may contain ε-steps. In fact, the definition remains the same - we must only extend the notions of 'stable node', 'initial steps', 'derivation' $g \xrightarrow{\sigma} h$. This is entirely straightforward:

4.2.9.1. DEFINITION. Let g be a process graph possibly containing ε.

(i) Let h be a subgraph of g, different from $\mathbb{0}$. Then $g \xrightarrow{\sigma} h$ if there is a path π from the root of g to h such that the proper steps in π determine the word $\sigma \in A^*$; *moreover, the last step of* π *must not be an* ε*-step.*

(ii) $g \xrightarrow{\sigma} \mathbb{0}$ is defined as before: if there is a path π determining σ to an end node, then $g \xrightarrow{\sigma} \mathbb{0}$ (even if the last step of π is an ε-step).

(iii) The set of initial steps of h, I(h), is defined as before in such a way that ε-steps are 'transparant' (like τ-steps).

Remark. The effect of (i) of this definition will be to declare endpoints of ε-steps to 'virtual nodes' that contribute no failure pairs to $F[\![g]\!]$.

4.2.9.2. DEFINITION. Let g be as in the previous definition. Let s be a node of g. Then s is a *stable* node if it is not an endpoint of an ε-step and there is no path leading from s starting with $\varepsilon^n\tau$ for some $n \geq 0$.

With these definitions, $F[\![g]\!]$ and $\equiv_F$ are defined as before in 4.2.2.

4.2.9.3. <u>EXAMPLES</u>. (i) Let g be $\to\circ\xrightarrow{\varepsilon}\circ$. Then $\mathcal{F}[\![g]\!] = \{(\lambda, \overline{\{\varepsilon\}}\ ), \varepsilon\}^{c}$. (Note that the root is a stable point.)

(ii) Let g be as in Figure 23 (a). Then $\mathcal{F}[\![g]\!] = \{\tau, (\lambda, \overline{\{\varepsilon\}}), \varepsilon\}^{c}$
The contribution $(\lambda, \overline{\{\varepsilon\}})$ is yielded by the endpoint of the τ-step, which is a stable point. Note that $\to\circ\xrightarrow{\tau}\circ$ has the same failure set, as should indeed be the case since $\tau + \varepsilon = \tau$ (this is equivalent with the second τ-law, T2, in Table 7).

(iii) Let g be as in Figure 23(b). Then $\mathcal{F}[\![g]\!] = \{(\lambda,\overline{\{a,\varepsilon\}}), (a, \overline{\{\varepsilon\}}), \varepsilon, a\varepsilon\}^{c}$. The displayed failure pairs are given by the root of g resp. the endpoint of the a-step.

Further, let h be as in Figure 23(c). Then $\mathcal{F}[\![h]\!] = \{(\lambda, \overline{\{\varepsilon\}}), (a, \overline{\{\varepsilon\}}), \varepsilon, a\varepsilon\}^{c}$. The displayed failure pairs are given by the endpoint of the τ-step resp. of the a-step.

Note that $g \not\equiv_{\mathcal{F}} h$, as should be the case; for, $\varepsilon + a = \tau + a$ yields a trace inconsistency: $(\varepsilon + a)\delta = (\tau + a)\delta \implies a\delta = \tau\delta + a\delta$.

(iv) In some cases, τ-steps may be replaced by ε-steps: let g,h be as in Figure 23(d),(e). Then

$$\mathcal{F}[\![g]\!] = \mathcal{F}[\![h]\!] = \{\tau, (a, \overline{\{\varepsilon\}}), a\varepsilon, (\lambda, \overline{\{\varepsilon\}}), \varepsilon\}^{c}.$$

Algebraically we have

$$\tau(a+\tau)+\tau = \tau(a+\tau)+\tau\tau = a+\tau\tau = a+\tau = a+\tau\varepsilon =$$
$$\tau(a+\varepsilon)+\tau\varepsilon = \tau(a+\varepsilon)+\tau.$$

(v) We have not admitted infinitely long ε-traces. Note that if we would have, the failure semantics of ε^{ω} (and of any process having only ε-steps and no finite branches) would be the empty set.

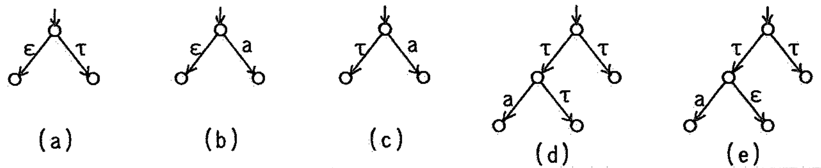


Figure 23 (a) (b) (c) (d) (e)

(vi) Also the failure semantics of the constant Δ can now be determined: Δ is interpreted in the model as the process graph (modulo $\equiv_{\mathcal{F}}$) g in Figure 24(a). Now $\mathcal{F}[\![g]\!] = \{\tau, \varepsilon\}$.

(vii) The failure semantics of $\Delta\delta$ has as representant the process graph h as in Figure 24(b) and also k as in Figure 24(c). Both have failure set $\{\tau\}$.

(viii) $\Delta\delta + \varepsilon$ is in the model (the equivalence class of) g as in Figure 24(d). Now $\mathcal{F}[\![g]\!] = \{\tau, \varepsilon\}$. (Cf. Example (vi).)

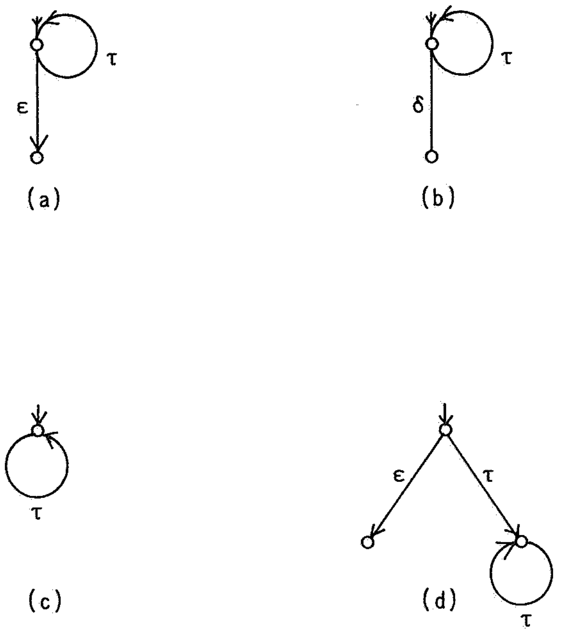
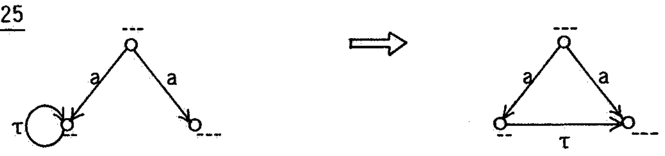


Figure 24

4.2.10. *The τ-jump transformation.*

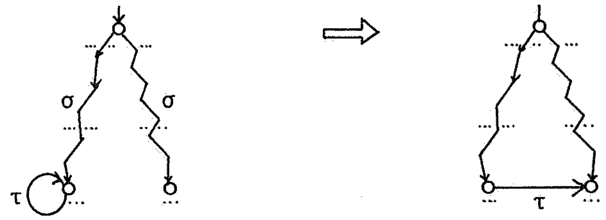
As in Section 3.5, we have a process graph transformation τ-jump, with the difference that the τ-loop authorizing the τ-jump may now be removed:

Figure 25



or more generally:

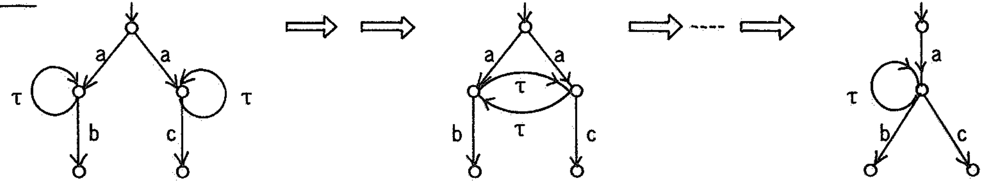
Figure 26



An example is already considered above: see Figure 21.

Note that, if *all* nodes with history σ are divergent, the present τ-jump does not help to get rid of this "σ-divergence", in accordance with Remark 4.2.8. (See Figure 27.)

Figure 27



4.3. *Characteristic processes.*

In this section we will consider characteristic processes, i.e. processes built exclusively from $\delta, \varepsilon, \tau, \Delta$ by means of + and $\cdot$; we will do this in the setting of bisimulation semantics with fair abstraction (as in Section 2) and failure semantics with fair abstraction from unstable divergence (as in Section 4.2).

4.3.1. *Characteristic processes in bisimulation semantics with fair abstraction.*

Since in this case $\Delta = \tau$, we need only consider processes built from $\delta, \varepsilon, \tau$. Now there is the following classification (we will not give the proof here):

4.3.1.1. PROPOSITION. *Each process involving only* δ- *and* τ-*steps is under bisimulation semantics equal to one of the processes*

$$\delta, \ \tau, \ \tau\delta, \ \tau\delta + \tau, \ \ \tau(\tau\delta + \tau).$$ □

We remark that this classification also pertains to infinite processes; and furthermore that these five processes constitute a process algebra.

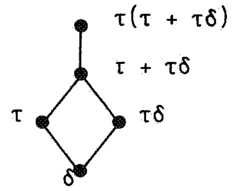
4.3.1.2. DEFINITION. Let x,y be elements of some process algebra. Then

$$x \sqsubseteq y \iff \exists z \ \ x + z = y.$$

(Note that $x \sqsubseteq y$ iff $x + y = y$.)

In this 'summand ordering', the five $\tau\delta$-processes are partially ordered as follows (see Figure 28):

Figure 28



After addition of ε there is a surprising fact, discovered by KOYMANS & VRANCKEN [16]: the classification, even of the finite processes over $\delta, \tau, \varepsilon$, then turns out to be an infinite one, and with the partial ordering just defined the finite $\delta\varepsilon\tau$-processes determine the same partial order as that of the *Rieger-Nishimura lattice* in intuitionistic propositional logic. (For an explanation of this structure see VAN DALEN [12].)

4.3.1.3. PROPOSITION (Koymans & Vrancken).

(i) All finite processes built from δ,ε,τ *in bisimulation semantics with fair abstraction are equal to one of the processes in the partial order of Figure 29.*

(ii) When infinite processes are also under consideration, a classification is obtained by completing the partial order in Figure 29 by two elements α *and* $\tau\alpha$ *where* α *is the sum of all the finite elements in the partial order.*

PROOF. KOYMANS & VRANCKEN [17]. □

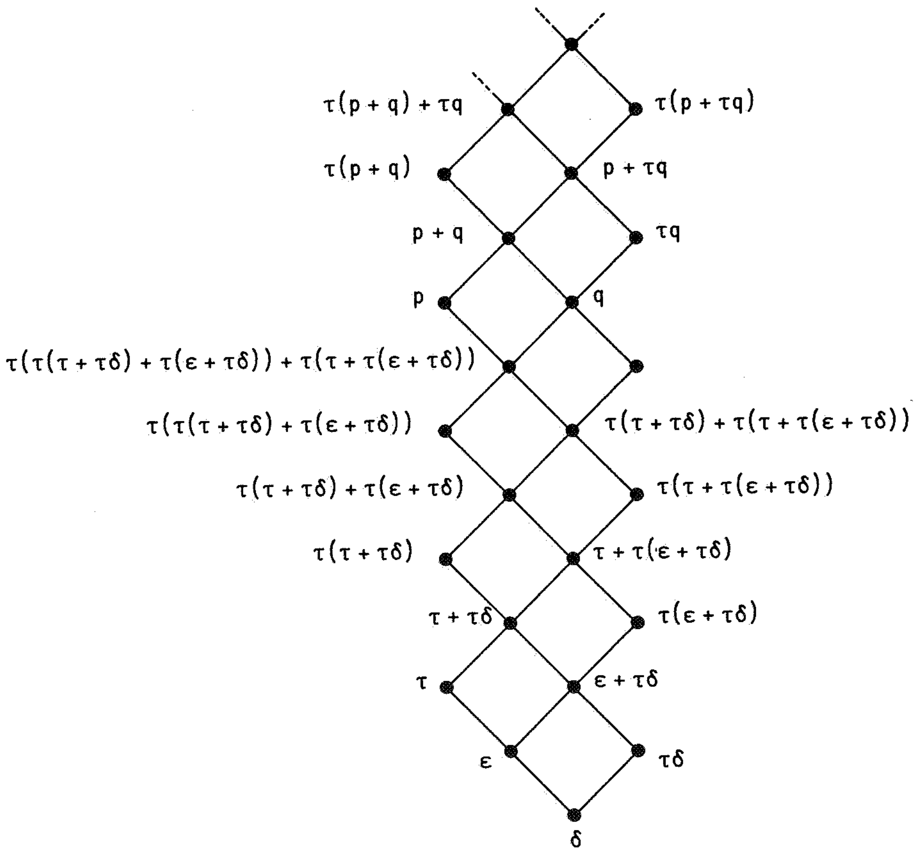

$\tau(p+q)+\tau q$
$\tau(p+\tau q)$
$\tau(p+q)$
$p+\tau q$
$p+q$
τq
p
q
$\tau(\tau(\tau+\tau\delta)+\tau(\varepsilon+\tau\delta))+\tau(\tau+\tau(\varepsilon+\tau\delta))$
$\tau(\tau(\tau+\tau\delta)+\tau(\varepsilon+\tau\delta))$
$\tau(\tau+\tau\delta)+\tau(\tau+\tau(\varepsilon+\tau\delta))$
$\tau(\tau+\tau\delta)+\tau(\varepsilon+\tau\delta)$
$\tau(\tau+\tau(\varepsilon+\tau\delta))$
$\tau(\tau+\tau\delta)$
$\tau+\tau(\varepsilon+\tau\delta)$
$\tau+\tau\delta$
$\tau(\varepsilon+\tau\delta)$
τ
$\varepsilon+\tau\delta$
ε
$\tau\delta$
δ

Figure 29

4.3.2. *Characteristic processes in failure semantics with abstraction from unstable divergence.*

When failure semantics is adopted, the infinitely many finite δετ-processes of 4.3.1 collaps to the five processes in Figure 30(a). When moreover Δ is added and the principle Δτ = τ (abstraction from unstable divergence) is adopted, only two new processes arise:

4.3.2.1. PROPOSITION. *In failure semantics with abstraction from unstable divergence, all processes built from* δ,ε,τ,Δ *are equal to one of the seven in the lattice of Figure 30(b). They form a process algebra with addition and multiplication tables as in Table 15, 16.*

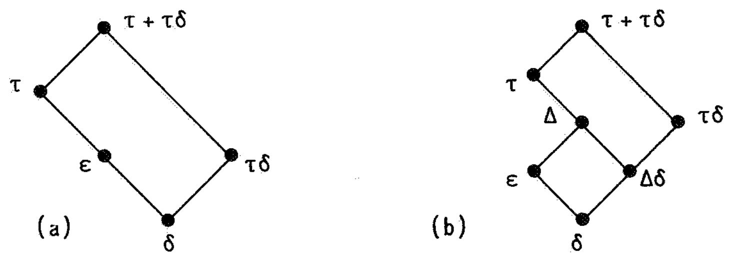


Figure 30

TABLE 15

+	δ	ε	τ	τδ	τδ+τ	Δ	Δδ
δ	δ	ε	τ	τδ	τδ+τ	Δ	Δδ
ε	ε	ε	τ	τδ+τ	τδ+τ	Δ	Δ
τ	τ	τ	τ	τδ+τ	τδ+τ	τ	τ
τδ	τδ	τδ+τ	τδ+τ	τδ	τδ+τ	τδ+τ	τδ
τδ+τ	τδ+τ	τδ+τ	τδ+τ	τδ+τ	τδ+τ	τδ+τ	τδ+τ
Δ	Δ	Δ	τ	τδ+τ	τδ+τ	Δ	Δ
Δδ	Δδ	Δ	τ	τδ	τδ+τ	Δ	Δδ

TABLE 16

•	δ	ε	τ	τδ	τδ+τ	Δ	Δδ
δ	δ	δ	δ	δ	δ	δ	δ
ε	δ	ε	τ	τδ	τδ+τ	Δ	Δδ
τ	τδ	τ	τ	τδ	τδ+τ	Δ	Δδ
τδ	τδ	τδ	τδ	τδ	τδ	τδ	τδ
τδ+τ	τδ	τδ+τ	τδ+τ	τδ	τδ+τ	τδ+τ	τδ
Δ	Δδ	Δ	τ	τδ	τδ+τ	Δ	Δδ
Δδ	Δδ	Δδ	Δδ	Δδ	Δδ	Δδ	Δδ

PROOF. The definition of failure semantics in Definition 4.2.2 shows that every characteristic process x must have a failure semantics $\mathcal{F}[\![x]\!]$ which has the form

$$\{\tau,\ (\lambda, \overline{\{\varepsilon\}}),\ (\lambda, \overline{\emptyset}),\ \varepsilon\}^{c} \cup \{\lambda\}$$

where some of the four displayed contributions in $\{\ldots\}^{c}$ may be absent. Accordingly, there are the following possibilities:

TABLE 17. Failure semantics of characteristic processes with abstraction of unstable divergence

x	$\mathcal{F}[\![x]\!] = \{\tau,\ (\lambda, \overline{\{\varepsilon\}}),\ (\lambda, \overline{\emptyset}),\ \varepsilon\}^{c} \cup \{\lambda\}$
	0 0 0 0
	--0-----0-----0-----1--
δ	0 0 1 0
	--0-----0-----1-----1--
	--0-----1-----0-----0--
ε	0 1 0 1
	==0=====1=====1=====0==
	==0=====1=====1=====1==
Δδ	1 0 0 0
Δ	1 0 0 1
τδ	1 0 1 0
τδ +τ	1 0 1 1
	--1-----1-----0-----0--
τ	1 1 0 1
	==1=====1=====1=====0==
	==1=====1=====1=====1==

The double erased possibilities (0110,...,1111) can be discarded since the second contribution $(\lambda, \overline{\{\varepsilon\}})$ is absorbed by the third, $(\lambda, \bar{\emptyset})$.

The single erased possibilities (0001,...,1100) can easily proved to be impossible.

Further, since we have not admitted infinite ε-traces, case 0000 can be discarded. (Otherwise, 0000, i.e. $\{\lambda\}$, would be a candidate for $F\llbracket\varepsilon^{\omega}\rrbracket$. A more natural definition would be $F\llbracket\varepsilon^{\omega}\rrbracket = \{\lambda, (\lambda, \bar{\emptyset})\}$, i.e. $\delta \underset{F}{=} \varepsilon^{\omega}$; this would be enforced by stipulating that, for g only consisting of nonterminating branches labeled with ε's, the set of initial steps $I(g) = \emptyset$.) □

4.3.2.2. PROPOSITION. *The set of seven characteristic processes from the previous proposition constitutes a final process algebra, 'final' in the sense that it cannot further be collapsed without trace inconsistency.*

PROOF. (1). $\Delta\delta = \tau\delta$ is impossible since $\tau = \tau + \Delta\delta$ and $\tau = \tau + \tau\delta$ is trace inconsistent.

(2). $\Delta = \varepsilon$ is impossible: $\Delta = \varepsilon \implies \Delta\tau = \varepsilon\tau \implies \Delta = \tau \implies$

$a\Delta(b+c) = a\tau(b+c) = a(b+c) = a(\Delta b + \Delta c) = a(\tau b + \tau c) = ab + ac.$

Now take $c = \delta$ to obtain a trace inconsistency.

(3). $\Delta\delta = \delta \implies \tau\Delta\delta = \tau\delta \implies \Delta\delta = \tau\delta$, trace inconsistent by (1).

(4). $\Delta = \Delta\delta \implies \tau x = \Delta\tau x = \Delta\delta\tau x = \Delta\delta \implies \tau a = \tau b$, trace inconsistent. □

4.3.2.3. QUESTION. Determine the lattice of characteristic processes in BS_{Δ} and in FS_{Δ}.

5. FAILURE SEMANTICS WITH CATASTROPHIC DIVERGENCE: FS_χ

In BROOKES, HOARE & ROSCOE [10] the principle of so-called 'catastrophic divergence' is adopted, stating that a divergent process should be set equal to the wholly arbitrary process CHAOS (in the terminology of [10]). A refinement of this failure semantics, still with catastrophic divergence, is presented in BROOKES & ROSCOE [11]: their version of the process CHAOS is different from the one in [10] in that it admits also many divergence possibilities. In both cases, all divergent processes are identified. An intuitive objection is that the divergent process Δa may diverge, but may also perform the a-action; whereas Δb will certainly not perform an a-action, and in this a priori respect the processes Δa, Δb are different. But, mathematically speaking, it is consistent to equate all divergent processes Δx, as [10] and [11] show; also in our context a process graph model can easily be manufactured satisfying $\Delta x = \Delta y$ for all x,y.

In the framework of the present paper it is simple to give an algebraic description of catastrophic divergence. If we adopt "for all x,y $\Delta x = \Delta y$" it makes sense to introduce a constant χ (for 'chaos') satisfying the axiom

$$\Delta x = \chi \qquad (*)$$

It follows that $\chi y = \chi$ (since $(\Delta x)y = \Delta(xy)$) and that $\chi + y = \chi$ (since $\chi + y = \Delta x + y = x + \Delta\delta + y = \Delta(x + y) = \chi$).

Using ε, we can go further and observe that $\chi = \Delta$, since $\Delta = \Delta\varepsilon = \chi$. This leads (by (*)) to the axiom $\Delta x = \Delta$; and this axiom is already implied by its instance

$$\Delta\delta = \Delta,$$

since then $\Delta x = \Delta\delta x = \Delta\delta = \Delta$.

We conclude, that starting from the basic axiom system for failure semantics FS_Δ in which nothing is assumed for divergence (except the rule DE^∞), and adopting equality of all divergent processes, we end up with the axiom system $FS_\Delta + \Delta\delta = \Delta$. In this axiom system, FS_χ for short, we have as derived equations

$$\Delta x = \Delta$$
$$\Delta + x = \Delta$$

and (hence) the rule for catastrophic divergence

$$DE^{\infty}_{\chi} \qquad \frac{\forall n \in \mathbb{N} \quad x_n = i_n.x_{n+1} + y_n, \qquad i_n \in I}{\tau_I(x_0) = \Delta}$$

We expect that a model $\mathcal{A}(FS_\chi)$ can be constructed along the lines discussed in Section 3.4 for failure semantics with explicit divergence. (Note that the remarks made there also apply to the present situation, i.e. either $\equiv_{\mathcal{F}}$ should be adapted to $\equiv^{w}_{\mathcal{F}}$ or the underlying domain of process graphs should be restricted.) What is interesting to us here, is that the constant Δ gives also for this option of catastrophic divergence a very easy description, in one equation: $\Delta\delta = \Delta$.

It is important to notice that FS_χ is *trace inconsistent with* FS: $\Delta\tau = \tau$ and $\Delta\delta = \Delta$ imply $\tau = \Delta\tau = \Delta\delta\tau = \Delta\delta = \Delta$, hence $ab = a\tau b = a\Delta\tau b = a\Delta = a\Delta\tau c = a\tau c = ac$. Similarly one proves that the union of FS_χ and BS are trace inconsistent.

5.1. REMARK. It is not hard to define a version of bisimulation semantics with catastrophic divergence and a corresponding model:

$$BS_\chi = BS_\Delta + \{\Delta x = \Delta + x = \Delta\}.$$

(Note that the equation $\Delta\delta = \Delta$ is now not sufficient, since the above derivation of $\Delta x = \Delta + x = \Delta$ used $\Delta x = x + \Delta\delta$, which is not valid in BS.) Clearly, FS_χ is an extension of BS_χ.

The construction of a model $\mathcal{A}(BS_\chi)$ is left to the reader.

5.2. REMARK. The lattice of characteristic processes is now as in Figure 31:

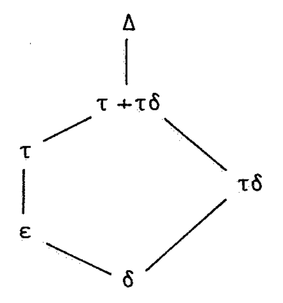


Figure 31

6. CONCLUDING REMARKS

We have considered the following process semantics:

TABLE 18. Process semantics with some characteristic equations.

BS_Δ	bisimulation semantics with explicit divergence	$\Delta\delta = \tau^\omega \neq \tau\delta$
BS	bisimulation semantics with fair abstraction of periodical divergence	$\Delta\delta = \tau^\omega = \tau\delta$, $\Delta = \tau$
BS_X	bisimulation semantics with catastrophic divergence	$\Delta = \Delta\delta = \tau^\omega \neq \tau\delta$ $\Delta + x = \Delta x = \Delta$
FS_Δ	failure semantics with explicit divergence	$\Delta\delta = \tau^\omega \neq \tau\delta$ $\Delta x = x + \Delta\delta$
FS	failure semantics with fair abstraction of unstable divergence	$\Delta\delta = \tau^\omega \neq \tau\delta$ $\Delta x = x + \Delta\delta$, $\Delta\tau = \tau$
FS_X	failure semantics with catastrophic divergence	$\Delta = \Delta\delta = \tau^\omega \neq \tau\delta$ $\Delta + x = \Delta x = \Delta$

Here BS_Δ is the most fundamental in the sense that it can be extended to the other ones. (See Figure 32.) In itself, BS_Δ, FS_Δ and BS_X are probably only of marginal interest. The interesting semantics are BS, FS and FS_X. They are mutually inconsistent (i.e. trace inconsistent).

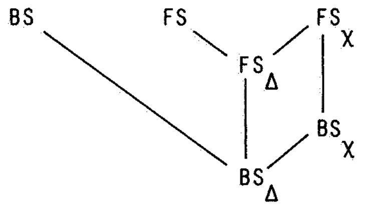


Figure 32

BS and FS_X are well-known process semantics; the first is known from Milner's work and the second from the work of Hoare e.a. [10]. The 'intermediate' semantics FS is, as far as we know, new.

The main merit of FS should be that it is a good tool for process verification (cf. BERGSTRA & KLOP [5] for an example of protocol verification using BS). One may expect that FS in some instances is even more suitable as verification medium than BS, since in FS much more identifications (between states of the process) are made, hence computations with processes in FS tend to be simpler than in BS.

FS seems definitely more suitable for verification purposes than FS_X, which has a too crude way of handling divergence. For instance, a protocol verification where one wants to abstract from a divergence (on the basis of certain fairness assumptions about the well-behaviour of possibly defective channels; cf. [5]) would simply be impossible if that divergence is treated as catastrophic.

The introduction of FS is in our view the primary contribution of the present paper; a secondary contribution is the use of the algebraic constant 'delay' Δ together with corresponding proof rules and axioms. Using Δ, we have obtained a uniform framework for the six process semantics discussed above.

This paper has not attempted an in-depth investigation of the new process semantics FS, with its main model $\mathcal{A}$(FS). We mention some directions for such an investigation in the following list of questions.

6.1. *Equating deadlock and livelock: some questions.*

The present view on BS_Δ and the extensions introduced in this paper, is by no means complete - several other interesting extensions of BS_Δ exist. One family of such extensions centers around 'readiness semantics'.

Also in the present framework there are some interesting extensions which are not explored here. One possibility is concerned with the equation

$$\Delta\delta = \tau\delta.$$

This equating of livelock and deadlock is a consequence of BS (see Proposition 2.1), but it is trace inconsistent with FS (see the proof of Proposition 4.3.2.2).

6.1.1. QUESTION. Is $\Delta\delta = \tau\delta$ consistent with FS_X? In FS_X this equation amounts to 'deadlock = chaos'.

It is clear that $BS_\Delta + \{\Delta\delta = \tau\delta\}$ is trace consistent, since $BS \vdash \Delta\delta = \tau\delta$. A model for $BS_\Delta + \{\Delta\delta = \tau\delta\}$ in which not yet $\Delta = \tau$ (as in BS is the case) is also easy to provide.

6.1.2. QUESTION. Is $FS_{\Delta} + \{\Delta\delta = \tau\delta\}$ trace consistent?

This axiom system $FS_{\Delta} + \{\Delta\delta = \tau\delta\}$ is remarkable because it is trace inconsistent with FS as well as BS (but maybe not with FS_{χ}). A noteworthy consequence is the following

6.1.3. PROPOSITION. $FS_{\Delta} + \{\Delta\delta = \tau\delta\} \vdash \Delta = \Delta\tau$.

PROOF. $\Delta x = x + \Delta\delta = x + \tau\delta \underset{*}{=} \tau x + \tau\delta = \tau x + \Delta\delta = \Delta\tau x$. Here equality '*' follows from axiom T4. □

Reversely, it does not seem to follow that $FS_{\Delta} + \{\Delta = \Delta\tau\} \vdash \Delta\delta = \tau\delta$. (We do have $FS_{\Delta} + \{\Delta = \Delta\tau\} \vdash \Delta\delta = \tau\delta + \Delta\delta$.) This opens the possibility that $FS_{\Delta} + \{\Delta = \Delta\tau\}$ is intermediate between FS_{Δ} and $FS_{\Delta} + \{\Delta\delta = \tau\delta\}$.

In any case, a complete classification of extensions of BS_{Δ} is at present not yet in sight. To obtain such a classification it will probably be important to study the process algebras consisting of characteristic terms (i.e. built from the constants $\delta,\varepsilon,\tau,\Delta$) in BS_{Δ}, FS_{Δ} and to determine their homomorphisms. (Cf. question 4.3.2.3.)

Still, the three mutually trace inconsistent axiom systems BS, FS, FS_{χ} considered in this paper may very well turn out to be the most important ones in this classification from the viewpoint of *abstraction of divergence*. We are not aware of any axiom system which might have as powerful abstraction facilities as these three.

6.2. *Completeness and maximality: some questions.*

In this section we mention some points of departure for further work on the topics of this paper.

6.2.1. QUESTION. Is the proof system for finite processes with failure semantics with τ and ε *complete* (w.r.t. the semantics defined in Section 3.2)? In BROOKES [9] a completeness result is proved for a similar situation, but the processes there have only one termination possibility (NIL).

6.2.2. QUESTION. Consider processes denoted by terms built from atoms a,b,c,..

by means of + and • and the constants $\delta,\varepsilon,\tau,\Delta$. So these are processes "in between" finite processes and regular processes (i.e. arising from finite but possibly cyclic process graphs); namely, these processes do not admit infinitely long or even arbitrarily long sequences of proper steps a,b,c,... To show that the proof system given in Section 4.1 is complete for these "finite processes with divergence". *)

6.2.3. QUESTION. As the previous question, now for regular processes. For regular processes under bisimulation semantics a complete proof system was given in BERGSTRA & KLOP [6], which was subsequently improved by MILNER [20]. With the usual additional syntax and axioms for μ-expressions (such as $\mu x.s(x) = s(\mu x.s(x))$) or systems of recursion equations, we conjecture that a complete proof system arises by combining the above proof systems with the axioms in MILNER [20].

This question can be considered both for failure semantics where divergence is explicit and the variant where $\Delta\tau = \tau$.

6.2.4. QUESTION. An important question is whether the failure semantics with $\Delta\tau = \tau$ is *maximal* in the sense that further extension leads to trace inconsistency. The analogous result for finite processes without τ (and a fortiori without Δ) was established in BERGSTRA, KLOP & OLDEROG [8].

A positive answer would be very satisfactory since it would mean that abstraction from unstable divergence (i.e. $\Delta\tau = \tau$) not only is trace consistent, but also the best one can get in failure semantics.

6.2.5. QUESTION. There are several theories intermediate between bisimulation semantics and failure semantics; an example is readiness semantics, another example is given by the 'forests' of ROUNDS [23]. It would be interesting to obtain information "up to what point" in the spectrum of process semantics it is possible to have $\Delta = \tau$.

We conclude with the obvious question:

6.2.6. QUESTION. Extend the axiomatisations in this paper to the context of ACP_τ, i.e. to the presence of parallel operators and communication.

*) A positive answer to this question as well as the previous one has been given by R. van Glabbeek (CWI Amsterdam).

REFERENCES

[1] BAETEN, J.C.M., BERGSTRA, J.A. & J.W. KLOP.
On the consistency of Koomen's Fair Abstraction Rule,
Report CS-R8511, Centrum voor Wiskunde en Informatica, Amsterdam 1985.

[2] DE BAKKER, J.W. & J.I. ZUCKER,
Processes and the denotational semantics of concurrency,
Information and Control 54 (1/2)(1982), p.70-120.

[3] BERGSTRA, J.A. & J.W. KLOP,
Algebra of Communicating Processes,
Report CS-R8421, Centrum voor Wiskunde en Informatica, Amsterdam 1984; to be published in:
Proc. of the CWI Symposium Mathematics and Computer Science (eds. J.W. de Bakker, M. Hazewinkel and J.K. Lenstra), North-Holland, Amsterdam.

[4] BERGSTRA, J.A. & J.W. KLOP,
Process algebra for synchronous communication,
Information and Control, Vol.60, Nos.1-3, p.109-137, 1984.

[5] BERGSTRA, J.A. & J.W. KLOP,
Verification of an alternating bit protocol by means of process algebra,
Report CS-R8404, Centrum voor Wiskunde en Informatica, Amsterdam 1984.

[6] BERGSTRA, J.A. & J.W. KLOP,
A complete inference system for regular processes with silent moves,
Report CS-R8420, Centrum voor Wiskunde en Informatica, Amsterdam 1984.

[7] BERGSTRA, J.A. & J.W. KLOP,
Algebra of communicating Processes with abstraction,
TCS 37 (1985) p.77-121.

[8] BERGSTRA, J.A., KLOP, J.W. & E.-R. OLDEROG,
Readies and failures in the Algebra of Communicating Processes,
Report CS-R8523, Centrum voor Wiskunde en Informatica, Amsterdam 1985.

[9] BROOKES, S.D.,
On the relationship of CCS and CSP,
in: Proc. 10th Int. Colloq. Automat. Lang. & Programming, Barcelona, (J. Díaz, Ed.), Springer LNCS 154, p.83-96, 1983.

[10] BROOKES, S., HOARE, C. & W. ROSCOE,
A Theory of Communicating Processes,
J. Assoc. Comput. Mach.31, No.3, p.560-599, 1984.

[11] BROOKES, S.D. & A.W. ROSCOE,
An improved failures model for communicating processes,
Technical Report Carnegie-Mellon University.
To appear in: Proc. of NSF-SERC Seminar on Concurrency, 1984, Springer LNCS, 1985.

[12] VAN DALEN, D.
Logic and Structure,
Springer 1980.

[13] HENNESSY, M.,
Synchronous and Asynchronous Experiments on Processes,
Report CSR-125-82, Univ. of Edinburgh, 1982.

[14] HOARE, C.A.R.,
Communicating Sequential Processes,
Comm. ACM 21 p.666-677, 1978.

[15] HOARE, C.A.R.,
A model for communicating sequential processes,
in: On the Construction of Programs" (R.M. McKeag and A.M. McNaghton, Eds.), p.229-243, Cambridge Univ. Press, London/New York, 1980.

[16] KOYMANS, C.J.P. & J.L.M. VRANCKEN,
Extending process algebra with the empty process ε,
Logic Group Preprint Series No.1, Department of Philosophy, Univ. of Utrecht, 1985.

[17] KOYMANS, C.J.P. & J.L.M. VRANCKEN,
Personal communication, May 1985.

[18] MILNER, R.,
A Calculus of Communicating Systems,
Springer LNCS 92, 1980.

[19] MILNER, R.,
Calculi for synchrony and asynchrony,
Theoret. Comput. Sci. 25 (1983), p.267-310.

[20] MILNER, R.,
A complete axiomatisation for observational congruence of finite-state behaviours,
manuscript, Univ. of Edinburgh, 1985.

[21] DE NICOLA, R. & M.C.B. HENNESSY,
Testing equivalences for processes,
TCS Vol.34, Nrs.1,2, p.83-133, 1984.

[22] OLDEROG, E.-R. & C.A.R. HOARE,
Specification-oriented semantics for communicating processes,
in: Proc. 10th ICALP, Barcelona, p.561-572, Springer LNCS 154 (1983); expanded version: Technical Monograph PRG-37, Oxford Univ. Comput. Lab., February 1984.

[23] ROUNDS, W.C.,
On the relationship between Scott domains, synchronization trees, and metric spaces,
Report CRL-TR-25-83, Univ. of Michigan, Ann Arbor, 1983.

[24] ROUNDS, W.C. & S.D. BROOKES,
Possible futures, acceptances, refusals, and communicating processes,
in: Proc. of 22nd IEEE Symposium on Foundations of Computer Science, Nashville, Tennessee (IEEE Computer Society Press, 1981) p.140-149.