ELSEVIER Theoretical Computer Science 170 (1996) 47–81

Theoretical Computer Science

Fundamental Study

Confluence for process verification

J.F. Groote[a], M.P.A. Sellink[b,*]

[a]*CWI, P.O. Box 94079, 1090 GB Amsterdam, Netherlands*
[b]*University of Amsterdam, Programming Research Group, Kruislaan 403, 1098 SJ Amsterdam, Netherlands*

Received September 1995; revised July 1996
Communicated by M. Nivat

Abstract

We provide several notions for confluence in processes and we show how these relate to τ-inertness, i.e. if $s \xrightarrow{\tau} s'$, then s and s' are equivalent. Using clustered linear processes we show how these notions can conveniently be used to reduce the size of state spaces and simplify the structure of processes while preserving equivalence.

Contents

1 Introduction 48
2 Preliminary definitions 49
3 Confluence and τ-inertness 50
  3.1 Strong confluence 51
  3.2 Weak confluence 52
  3.3 Weak bisimulation confluence 54
4 Other graph equivalences 58
  4.1 Two definitions generalised 58
  4.2 Strong bisimulation 59
  4.3 Branching bisimulation 59
  4.4 Finite trace equivalence 60
5 Transition systems that are not τ-well-founded 60
  5.1 Progressing and non-progressing τ-steps 61
  5.2 Progressing confluence 61
6 Confluence of linear processes 66
  6.1 Linear processes 66
  6.2 A condition for strong confluence 67
  6.3 A condition for weak progressing confluence 68
7 State space reduction 69
8 Two examples 73
  8.1 Concatenation of two queues 74
  8.2 The alternating bit protocol 75
Acknowledgements 80
References 81

* Corresponding author. E-mail: alex@fwi.uva.nl or jfg@cwi.nl.

1. Introduction

In his seminal book [16] Milner devotes a chapter to the notions strong and observation confluence in process theory. Many other authors have confirmed the importance of confluence. For example, in [17, 13, 8] the notion is used for on the fly reduction of finite state spaces and in [16, 19] it has been used for the verification of protocols.

We feel that a more general treatment of the notion of confluence is in order. The first reason for this is that the treatment of confluence has always been somewhat ad hoc in the setting of process theory. This strongly contrasts with for instance term rewriting [14], where confluence is one of the major topics. In particular, we want to clarify the relation with *τ-inertness*, which says that if $s \xrightarrow{\tau} s'$, then s and s' are equivalent in some sense.

The second reason is that we want to develop systematic ways to prove distributed systems correct in a precise and formal fashion. In this way we want to provide techniques to construct fault free distributed systems. For this purpose the language μCRL is designed, being process algebra extended with data [10]. In [9] a proof theory has been developed and in [18] it has been shown how correctness proofs can be checked using proof checkers, giving us the means to deliver descriptions of distributed systems with the highest thinkable level of correctness. In order to show the applicability of the techniques several protocols have been verified [15, 12, 6], both from theoretical and practical perspectives. Experience with these protocols gave rise to the development of new and the adaption of existing techniques to make systematic verification possible [7, 6]. Employing confluence also belongs to these techniques. It appears to enable easier verification of distributed systems, which in essence boils down to the application of τ-inertness.

In the first part of the paper we address the relationship between confluence and τ-inertness. The notions proposed in [16] all imply τ-inertness. We come up with a more general notion, namely weak $\sim$-confluence where $\sim$ is some equivalence. For several notions of equivalence we show that weak $\sim$-confluence and τ-inertness with respect to $\sim$ coincide (especially for weak and branching bisimulation), provided the process is τ-well-founded (there are no infinite τ-paths).

However, there are many protocols, where there are infinite τ-paths, for instance communication protocols over unreliable channels. Therefore, we introduce the distinction between progressing and non-progressing τ's and show that weakly progressing confluence is enough to guarantee τ-inertness. Contrary to what one would expect, weakly $\sim$-progressing confluence does not imply τ-inertness.

In the second part of this paper we direct our attention to establishing confluence. It hardly makes sense to establish confluence directly on transition systems, because these are generally far too large to be represented. Therefore, we try to establish confluence on Linear Processes [7] which represent large transition systems symbolically in a compact way.

In the third part we show how we can use τ-inertness to reduce state spaces and carry out verifications on linear processes. We provide two examples illustrating that the application of confluence often reduces the size of state spaces considerably and simplifies the structure of distributed systems, while preserving branching bisimulation. This is in general a very profitable preprocessing step before analysis, testing or simulation of a distributed system.

2. Preliminary definitions

Let S and S' be sets. A binary relation R on S and S' is a subset of $S \times S'$. We write xRy for $(x,y) \in R$. If $S \equiv S'$ we say that R is a binary relation on S. We write $xRyRz$ for $xRy \wedge yRz$ and R^* for the reflexive, transitive closure of R. The symmetric closure of R is denoted by R°. We write $R^{\circledast}$ for $(R^\circ)^*$. This relation is an equivalence relation, since symmetry is preserved[1] by $(-)^*$. Finally, we write Δ_R for the diagonal relation on R, i.e. $\Delta_R = \{(x,x) \mid x \in R\}$.

Definition 2.1. A binary relation R on S is called well-founded iff there is no infinite sequence $\{s_i\}_{i=0}^{\infty}$ such that $s_{i+1}Rs_i$ for all $i \in \mathbb{N}$.

Let Act be an arbitrary set of actions, containing a distinguished element τ. Throughout this paper we fix this set of actions, except that in Section 5 and further we distinguish progressing τ-steps (denoted by $\tau_>$) and non-progressing τ-steps (denoted by $\tau_<$).

Definition 2.2. A transition system is a pair $(S, \longrightarrow\!\!\triangleright)$ with S a set of states and $\longrightarrow\!\!\triangleright \subseteq S \times \text{Act} \times S$. We write $s \xrightarrow{a}\!\!\triangleright\, t$ instead of $\longrightarrow\!\!\triangleright\,(s,a,t)$.

Note that this definition implies that we exclude transition systems that have more than one a-step from s to s'.

Convention 2.3. *We introduce the following notations* $(a \in \text{Act}, \sigma \in \text{Act}^\star)$*:*

- $s \xrightarrow{a^*}\!\!\triangleright\, t$ *means* $s \equiv u_1 \xrightarrow{a}\!\!\triangleright \cdots \xrightarrow{a}\!\!\triangleright\, u_n \equiv t$ *for some* $u_1,\ldots,u_n$ *with* $n \geqslant 1$,
- $s \xrightarrow{\sigma a}\!\!\triangleright\, t$ *means* $s \xrightarrow{\sigma}\!\!\triangleright\, u \xrightarrow{a}\!\!\triangleright\, t$ *for some* u,
- $s \overset{a}{=\!=\!\!\triangleright}\, t$ *means* $s \xrightarrow{\tau^*}\!\!\triangleright\, u_1 \xrightarrow{a}\!\!\triangleright\, u_2 \xrightarrow{\tau^*}\!\!\triangleright\, t$ *for some* u_1, u_2.
- $s \overset{a}{\equiv\!\equiv\!\!\triangleright}\, t$ *means* $s \overset{a}{=\!=\!\!\triangleright}\, t \vee (a \equiv \tau \wedge s \xrightarrow{\tau^*}\!\!\triangleright\, t)$.

Definition 2.4. A relation $R \subseteq S \times S'$ is called a weak bisimulation on $(S, \longrightarrow\!\!\triangleright)$ and $(S', \longrightarrow\!\!\blacktriangleright)$ iff

$$sRs' \Longrightarrow \begin{cases} s \xrightarrow{a}\!\!\triangleright\, t & \Longrightarrow \exists t'.\ s' \overset{a}{\equiv\!\equiv\!\!\blacktriangleright}\, t' \wedge tRt' \\ s' \xrightarrow{a}\!\!\blacktriangleright\, t' & \Longrightarrow \exists t.\ s \overset{a}{\equiv\!\equiv\!\!\triangleright}\, t \wedge tRt' \end{cases}$$

for all $s \in S$ and for all $s' \in S'$.

[1] The converse does not hold, $(R^*)^\circ$ is not necessarily transitive.

We say that R is a weak bisimulation on $(S, \mathrel{-\!\!\!\rhd})$ iff R is a weak bisimulation on $(S, \mathrel{-\!\!\!\rhd})$ and $(S, \mathrel{-\!\!\!\rhd})$.

Let $\{R_i\}_{i \in I}$ be a collection of weak bisimulations on $(S, \mathrel{-\!\!\!\rhd})$ and $(S', \mathrel{-\!\!\!\blacktriangleright})$, then $\bigcup_{i \in I} R_i$ is also a weak bisimulation on $(S, \mathrel{-\!\!\!\rhd})$ and $(S', \mathrel{-\!\!\!\blacktriangleright})$. Consequently there exists a maximal weak bisimulation on $(S, \mathrel{-\!\!\!\rhd})$ and $(S', \mathrel{-\!\!\!\blacktriangleright})$, namely the union of all weak bisimulations on $(S, \mathrel{-\!\!\!\rhd})$ and $(S', \mathrel{-\!\!\!\blacktriangleright})$. This maximal weak bisimulation is denoted as $\underline{\leftrightarrow}_w$. If $s \mathrel{\underline{\leftrightarrow}_w} t$ then we say that s and t are *weakly bisimilar*. $\underline{\leftrightarrow}_w$ is an equivalence relation.

Definition 2.5. A transition system $(S, \mathrel{-\!\!\!\rhd})$ is a-well-founded iff $\{(s,t) \mid t \overset{a}{\mathrel{-\!\!\!\rhd}} s\}$ is a well-founded relation on S, i.e. there is no infinite sequence of the form

$$s_1 \overset{a}{\mathrel{-\!\!\!\rhd}} s_2 \overset{a}{\mathrel{-\!\!\!\rhd}} s_3 \overset{a}{\mathrel{-\!\!\!\rhd}} \cdots\cdots$$

Let $\mathcal{O}$ be the class of ordinals.

Definition 2.6. Let $R \subseteq S \times S$ be a well-founded relation on S. We define a mapping d_R: $S \longrightarrow \mathcal{O}$ as follows:

$$d_R(t) = \sup\{1 + d_R(s) \mid sRt\}.$$

If $R = \{(x,y) \mid y \overset{a}{\mathrel{-\!\!\!\rhd}} x\}$ then we write d_a instead of d_R. $d_a(t)$ is the length of the longest chain of a-steps starting from t.

3. Confluence and τ-inertness

In this section we introduce three different notions of *confluence*, namely strong confluence, weak confluence and weak $\underline{\leftrightarrow}_w$-confluence. We investigate whether or not the different notions of confluence are sufficiently strong to serve as a condition for

$$s \overset{\tau}{\mathrel{-\!\!\!\rhd}} t \Longrightarrow s \mathrel{\underline{\leftrightarrow}_w} t \tag{1}$$

to hold. It is obvious that (1) is not valid in general. A simple counterexample is

$$\begin{tikzcd} s \arrow[r, "\tau"] \arrow[d, "a"'] & s' \\ t & \end{tikzcd} \tag{2}$$

where a is not a τ-step. Transition systems that satisfy (1) are called *τ-inert with respect to $\underline{\leftrightarrow}_w$*.

3.1. Strong confluence

In this subsection we prove that strongly confluent transition systems are τ-inert with respect to $\underline{\leftrightarrow}_w$. Although this is actually a well-known result we work out the proof in full detail because it nicely illustrates the technique that is used throughout this paper, namely to define a relation R satisfying $\{(x,y) \mid x \xrightarrow{\tau} y\} \subseteq R$ and show that R is a weak bisimulation. Using this technique leads – in our opinion – to elegant proofs that are easy to understand. Many of the results of this work can also be proved, however, by applying (an extension of) the work of Arnold and Dicky [2].

Definition 3.1. A transition system $(S, \longrightarrow)$ is called strongly confluent for a iff for each pair $s \xrightarrow{a} t$ and $s \xrightarrow{\tau} s'$ of different steps there exists a state t' such that $t \xrightarrow{\tau} t'$ and $s' \xrightarrow{a} t'$. In a diagram:

$$\begin{tikzcd}
s \arrow[r, "\tau"] \arrow[d, "a"'] & s' \arrow[d, dashed, "a"'] \\
t \arrow[r, dashed, "\tau"] & t'
\end{tikzcd}$$

A transition system $(S, \longrightarrow)$ is called strongly confluent iff it is strongly confluent for a, for all $a \in \textsc{Act}$.

Omitting the word "different" in Definition 3.1 would give a stronger notion:

$$s \xrightarrow{\qquad\tau\qquad} t$$

is strongly confluent, but would not be strongly confluent if the word "different" was omitted.

Theorem 3.2. *Strongly confluent transition systems are τ-inert with respect to $\underline{\leftrightarrow}_w$.*

Proof. Let $(S, \longrightarrow)$ be a strongly confluent transition system. Consider

$$R = \{(x,y) \mid x \xrightarrow{\tau} y\} \cup \Delta_S.$$

It suffices to prove that R is a weak bisimulation on $(S, \longrightarrow)$, as in that case, by definition, $R \subseteq \underline{\leftrightarrow}_w$. So $s \xrightarrow{\tau} t \Longrightarrow sRt \Longrightarrow s \,\underline{\leftrightarrow}_w\, t$. Assume sRs'. We have to prove:

(i) $s \xrightarrow{a} t \Longrightarrow \exists t'.\ s' \stackrel{a}{\Longrightarrow} t' \wedge tRt'$

(ii) $s' \xrightarrow{a} t' \Longrightarrow \exists t.\ s \stackrel{a}{\Longrightarrow} t \wedge tRt'$

If $s \equiv s'$ then we are trivially done. So assume $s \xrightarrow{\tau} s'$.

Proof of (i): Assume $s \xrightarrow{a} t$. The situation is now as follows:

$$\begin{tikzcd}
s \arrow[r, "\tau"] \arrow[d, "a"'] & s' \\
t &
\end{tikzcd}$$

If these two transitions are different, then there exists a state t' such that $s' \xrightarrow{a} t'$ and $t \xrightarrow{\tau} t'$, by the definition of strong confluence. If $t \xrightarrow{\tau} t'$ then tRt' so (i) is satisfied. If these two transitions $s \xrightarrow{a} t$ and $s \xrightarrow{\tau} s'$ are identical, we have $t \equiv s'$ and $a \equiv \tau$, so tRs' by $\Delta_S \subseteq R$. Again, (i) is satisfied.

Proof of (ii): Assume $s' \xrightarrow{a} t'$. The situation is then as follows:

$$\begin{tikzcd}
s \arrow[r, "\tau"] & s' \arrow[d, "a"'] \\
& t'
\end{tikzcd}$$

Now take $t \equiv t'$. This does the job, since $s \xrightarrow{\tau a} t'$ and $t'Rt'$ since $\Delta_S \subseteq R$. □

The converse of Theorem 3.2 is obviously not valid. A transition system that is τ-inert with respect to $\leftrightarrows_w$ is not necessarily strongly confluent. As a counterexample one can take (2) with $a \equiv \tau$. This counterexample means that strong confluence is actually a stronger notion than we need since we are primarily interested in τ-inertness (w.r.t. $\leftrightarrows_w$). Hence we introduce a weaker notion of confluence, which differs from strong confluence in that we allow τ-steps in the paths from t to t' and from s' to t'.

3.2. Weak confluence

Definition 3.3. A transition system $(S, \longrightarrow)$ is called weakly confluent for a iff for each pair $s \xrightarrow{a} t$ and $s \xrightarrow{\tau} s'$ of different steps there exists a state t' such that $s' \xRightarrow{a} t'$ and $t \xrightarrow{\tau^*} t'$. In a diagram:

$$\begin{tikzcd}
s \arrow[r, "\tau"] \arrow[d, "a"'] & s' \arrow[d, Rightarrow, "a"'] \\
t \arrow[r, dashed, "\tau^*"] & t'
\end{tikzcd}$$

A transition system $(S, \longrightarrow)$ is called weakly confluent iff it is weakly confluent for a, for all $a \in \textsc{Act}$.

The following example (due to Roland Bol) shows that weak confluence is too weak to serve as a condition for (1) to hold, i.e. weak confluent transition systems are not necessarily τ-inert with respect to $\underline{\leftrightarrow}_w$.

$$\begin{tikzcd}
s \arrow[r, shift left, "\tau"] \arrow[d, "a"'] & s' \arrow[l, shift left, "\tau"] \arrow[d, "\tau"] \\
t & t'
\end{tikzcd} \tag{3}$$

This transition system is weakly confluent but $s' \xrightarrow{\tau} t'$ does not connect bisimilar states if $a \not\equiv \tau$. Note that (3) is not τ-well-founded. In Theorem 3.5 we prove that τ-well-founded, weakly confluent transition systems are τ-inert with respect to $\underline{\leftrightarrow}_w$. This means that we cannot replace (3) by a τ-well-founded counterexample.

The following lemma is very useful and frequently used in the remaining of the paper.

Lemma 3.4. *Let $(S, \longrightarrow)$ be τ-well-founded and weakly confluent for τ. Let $s \xrightarrow{\tau^*} s'$ and $s \xrightarrow{\tau^*} t$, then there exists a t' such that $s' \xrightarrow{\tau^*} t'$ and $t \xrightarrow{\tau^*} t'$. In a diagram:*

$$\begin{tikzcd}
s \arrow[r, "\tau^*"] \arrow[d, "\tau^*"'] & s' \arrow[d, dashed, "\tau^*"] \\
t \arrow[r, dashed, "\tau^*"] & t'
\end{tikzcd}$$

Proof. We use induction on $d_\tau(s)$. Note that the lemma is trivial if $s \equiv s'$ or $s \equiv t$. (In particular, the lemma is trivial for $d_\tau(s) = 0$.) Suppose that $s \not\equiv s'$ and $s \not\equiv t$, say $s \xrightarrow{\tau} u \xrightarrow{\tau^*} t$ and $s \xrightarrow{\tau} u' \xrightarrow{\tau^*} s'$. Assume that the lemma holds for all x satisfying $d_\tau(x) < d_\tau(s)$. We distinguish two cases:

- $s \xrightarrow{\tau} u$ and $s \xrightarrow{\tau} u'$ are identical. We can draw the following picture:

$$\begin{tikzcd}
s \arrow[r, "\tau"] & u \equiv u' \arrow[r, "\tau^*"] \arrow[d, "\tau^*"'] & s' \arrow[d, "\tau^*"] \\
& t \arrow[r, "\tau^*"] & t'
\end{tikzcd}$$

$t \xrightarrow{\tau^*} t'$ and $s' \xrightarrow{\tau^*} t'$ exist because the lemma holds (by induction) in u.

- $s \xrightarrow{\tau^*} u$ and $s \xrightarrow{\tau^*} u'$ are different. Now we can draw the following picture:

$$\begin{tikzcd}
s \arrow[r, "\tau"] \arrow[d, "\tau"'] & u' \arrow[r, "\tau^*"] \arrow[d, "\tau^*"'] & s' \arrow[d, "\tau^*"'] \\
u \arrow[r, "\tau^*"] \arrow[d, "\tau^*"'] & u'' \arrow[r, "\tau^*"] \arrow[d, "\tau^*"'] & s'' \arrow[d, "\tau^*"'] \\
t \arrow[r, "\tau^*"] & t'' \arrow[r, "\tau^*"] & s'
\end{tikzcd}$$

The upper-left part of the diagram is given by weak confluence for τ. The other parts are given by induction hypotheses for u, u' and u''. □

We cannot omit "τ-well-foundedness" as a condition in Lemma 3.4. As a counter-example take (3) with $a \equiv \tau$.

The following relations form the core of all our remaining "confluence implies τ-inertness" proofs.

$$\mathcal{T} \triangleq \{(x,y) \mid x \xrightarrow{\tau} y\}$$
$$B_w \triangleq \{(x,y) \mid x \xrightarrow{\tau} y \vee x \leftrightarroweq_w y\}.$$

Theorem 3.5. *Let $(S, \longrightarrow)$ be τ-well-founded. If $(S, \longrightarrow)$ is weakly confluent then $(S, \longrightarrow)$ is τ-inert with respect to $\leftrightarroweq_w$.*

Proof. This theorem can be proved directly by showing that $\mathcal{T}^*$ is a weak bisimulation on $(S, \longrightarrow)$. We omit this proof here because the result also follows from Theorem 3.9. The interested reader can find the proof in [11]. □

In Section 3.1 we mentioned that τ-inertness with respect to $\leftrightarroweq_w$ does not imply strong confluence. The same (τ-well-founded) counterexample – Diagram (2) with $a \equiv \tau$ – illustrates that τ-inertness with respect to $\leftrightarroweq_w$ does not even imply weak confluence. So weak confluence and τ-inertness with respect to $\leftrightarroweq_w$ are independent notions. If we restrict ourselves to the τ-well-founded transition systems we have strict inclusion of the weakly confluent transition systems into the transition systems that are τ-inert with respect to $\leftrightarroweq_w$.

3.3. Weak bisimulation confluence

In the definition below we introduce yet another notion of confluence. This third notion is optimal in the sense that it is equivalent with τ-inertness with respect to $\leftrightarroweq_w$ for τ-well-founded systems.

Definition 3.6. A transition system $(S, \longrightarrow)$ is called weakly $\underline{\leftrightarrow}_w$-confluent for a iff for each pair $s \xrightarrow{a} t$ and $s \xrightarrow{\tau} s'$ of different steps there exist states u and u' such that $u \, \underline{\leftrightarrow}_w \, u'$, $s' \overset{a}{\Longrightarrow} u'$ and $t \xrightarrow{\tau^*} u$. In a diagram:

$$\begin{tikzcd}
s \arrow[r, "\tau"] \arrow[d, "a"'] & s' \arrow[d, Rightarrow, "a"'] \\
t \arrow[r, dashed, "\tau^*"] & u \, \underline{\leftrightarrow}_w \, u'
\end{tikzcd}$$

A transition system $(S, \longrightarrow)$ is called weakly $\underline{\leftrightarrow}_w$-confluent iff it is weakly $\underline{\leftrightarrow}_w$-confluent for a, for all $a \in \textsc{Act}$.

We want to prove that in τ-well-founded transition systems "weak $\underline{\leftrightarrow}_w$-confluence" implies "$\tau$-inertness with respect to $\underline{\leftrightarrow}_w$". Clearly, this immediately implies Theorem 3.5. In [11], however, we proved Theorem 3.5 directly by showing that $\mathcal{T}^*$ is a weak bisimulation on weakly confluent transition systems. Since $\mathcal{T}^*$ is not necessarily a weak bisimulation on weakly $\underline{\leftrightarrow}_w$-confluent transition systems – as example (4) below shows – we cannot use $\mathcal{T}^*$ to prove Theorem 3.9.

$$\begin{tikzcd}
s \arrow[r, "\tau"] \arrow[d, "a"'] & t \arrow[d, "a"'] \\
s' & t'
\end{tikzcd} \tag{4}$$

Here $\mathcal{T}^* = \{(s,s),(s',s'),(s,t),(t,t),(t',t')\}$ is not a weak bisimulation although the transition system is weakly $\underline{\leftrightarrow}_w$-confluent since $s' \xrightarrow{\tau^*} s' \, \underline{\leftrightarrow}_w \, t'$. The diagram above is also an example of a transition system that is weakly $\underline{\leftrightarrow}_w$-confluent but not weakly confluent.

The following lemma is used in the proof of Lemma 3.8.

Lemma 3.7. *Let $(S, \longrightarrow)$ be τ-well-founded and weakly $\underline{\leftrightarrow}_w$-confluent. Let $s \xrightarrow{\tau^*} s'$ and $s \overset{a}{\Longrightarrow} t$. There exists a state t' such that $tB_w^{\circledast}t'$ and $s' \overset{a}{\Longrightarrow} t'$. In a diagram:*

$$\begin{tikzcd}
s \arrow[r, "\tau^*"] \arrow[d, Rightarrow, "a"'] & s' \arrow[d, Rightarrow] \\
t \arrow[r, dash, dashed, "B_w^{\circledast}"] & t'
\end{tikzcd}$$

Proof. We use induction on $d_\tau(s)$. If $s \equiv s'$ (and in particular if $d_\tau(s) = 0$) then the lemma is trivial. Suppose that $s \xrightarrow{\tau} s'' \xrightarrow{\tau^*} s'$ and assume that the lemma holds for all states x such that $d_\tau(x) < d_\tau(s)$. If $a \equiv \tau$ then $t' \equiv s'$ satisfies the required properties, namely $tB_w^{\circledast}t'$ since $t \xleftarrow{\tau^*} s \xrightarrow{\tau} s'$ and $s' \xRightarrow{\tau} t'$ by reflexivity of $\xRightarrow{\tau}$.

Assume that $a \not\equiv \tau$. Say $s \xrightarrow{\tau^*} v \xrightarrow{a} u \xrightarrow{\tau^*} t$. We distinguish two cases:

- $v \equiv s$. We can draw the following picture:

$$\begin{tikzcd}
s \arrow[r, "\tau"] \arrow[d, "a"'] & s'' \arrow[r, "\tau^*"] \arrow[d, Rightarrow, "a"'] & s' \arrow[d, Rightarrow, "a"'] \\
u \arrow[r, "\tau^*"] \arrow[d, "\tau^*"'] & u'' \underline{\leftrightarrow}_w u' \arrow[r, dash, "B_w^{\circledast}"] & t' \\
t & &
\end{tikzcd}$$

The left part of the diagram is given by weak $\underline{\leftrightarrow}_w$-confluence and the right part of the diagram follows from applying the induction hypothesis on s'', which is allowed since $d_\tau(s'') < d_\tau(s)$. Now $tB_w^{\circledast}t'$ and $s' \xRightarrow{a} t'$ so t' satisfies the required properties.

- $v \not\equiv s$. We are in the following situation:

$$\begin{tikzcd}
s \arrow[r, "\tau"] \arrow[d, "\tau"'] & s' \arrow[d, "\tau^*"'] \\
v \arrow[r, "\tau^*"] \arrow[d, Rightarrow, "a"'] & v' \arrow[d, Rightarrow, "a"'] \\
t \arrow[r, dash, "B_w^{\circledast}"] & t'
\end{tikzcd}$$

The upper part of the diagram follows directly from Lemma 3.4 and the lower part of the diagram is given by application of the induction hypothesis on v, which is allowed since $d_\tau(v) < d_\tau(s)$. We are done because $tB_w^{\circledast}t'$ and $s' \xRightarrow{a} t'$. $\square$

Lemma 3.8. *Let $(S, \longrightarrow)$ be τ-well-founded and weakly $\underline{\leftrightarrow}_w$-confluent, then the equivalence relation $B_w^{\circledast}$, defined in Section 3.2, is a weak bisimulation on $(S, \longrightarrow)$.*

Proof. As $sB_w^{\circledast}s'$, we may assume that there exists an $n \geqslant 0$ such that

$$s \equiv x_0 B_w^{\circ} x_1 B_w^{\circ} \cdots\cdots B_w^{\circ} x_{n-1} B_w^{\circ} x_n \equiv s'.$$

We have to show:

(i) $s \xrightarrow{a} t \Longrightarrow \exists t'.s' \overset{a}{\Longrightarrow} t' \wedge tB_w^{\circledast}t'$

(ii) $s' \xrightarrow{a} t' \Longrightarrow \exists t.s \overset{a}{\Longrightarrow} t \wedge tB_w^{\circledast}t'$

Since $B_w^{\circledast}$ is symmetric we have (i) $\Longleftrightarrow$ (ii). We prove (i).

Proof of (i): We prove the following slightly stronger result by induction to n:

$$s \overset{a}{\Longrightarrow} t \Longrightarrow \exists t'.s' \overset{a}{\Longrightarrow} t' \wedge tB_w^{\circledast}t'. \qquad (\mathrm{i}')$$

For $n = 0$ the validity of (i′) is trivial. Let $n > 0$ and assume that the lemma holds for $n - 1$. Consider the following diagram:

$$\begin{tikzcd}
x_0 \arrow[r, dash, "B_w^{\circledast}"] \arrow[d, Rightarrow, "a"'] & x_{n-1} \arrow[r, dash, "B_w^{\circ}"] \arrow[d, Rightarrow, "a"'] & x_n \\
t \arrow[r, dash, "B_w^{\circledast}"] & u &
\end{tikzcd}$$

u is given by the induction hypothesis. We distinguish three cases for $x_{n-1}B_w^{\circ}x_n$:

$$\begin{tikzcd}
x_0 \arrow[r, dash, "B_w^{\circledast}"] \arrow[d, Rightarrow, "a"'] & x_{n-1} \arrow[r, "\tau"] \arrow[d, Rightarrow, "a"'] & x_n \\
t \arrow[r, dash, "B_w^{\circledast}"] & u &
\end{tikzcd}$$

Applying Lemma 3.7 gives a state t' satisfying the right properties. If $x_{n-1} \underline{\leftrightarrow}_w x_n$ then use the definition of weak bisimulation and if $x_{n-1} \xleftarrow{\tau} x_n$ then simply take $t' \equiv u$. □

Theorem 3.9. *Let* $(S, \longrightarrow)$ *be* τ*-well-founded. Then* $(S, \longrightarrow)$ *is weakly* $\underline{\leftrightarrow}_w$*-confluent iff* $(S, \longrightarrow)$ *is* τ*-inert with respect to* $\underline{\leftrightarrow}_w$.

Proof. Let $(S, \longrightarrow)$ be τ-well-founded.

($\Longrightarrow$) By Lemma 3.8, the equivalence relation $B_w^{\circledast}$, defined in Section 3.2, is a weak bisimulation. Since $\underline{\leftrightarrow}_w$ is the union of all weak bisimulations, we have $B_w^{\circledast} \subseteq \underline{\leftrightarrow}_w$. Now $s \xrightarrow{\tau} t \Longrightarrow sB_w^{\circledast}t \Longrightarrow s \underline{\leftrightarrow}_w t$.

($\Longleftarrow$) Let $s \xrightarrow{a} t$ and $s \xrightarrow{\tau} s'$. Then $s \underline{\leftrightarrow}_w s'$ since $(S, \longrightarrow)$ satisfies (1). Now, by definition, there exists a state t' such that $s' \overset{a}{\Longrightarrow} t'$ and $t \underline{\leftrightarrow}_w t'$, so we are done. □

Note that we need the τ-well-foundedness of $(S, \longrightarrow)$ only in the proof from left to right (it allows us to apply Lemma 3.8).

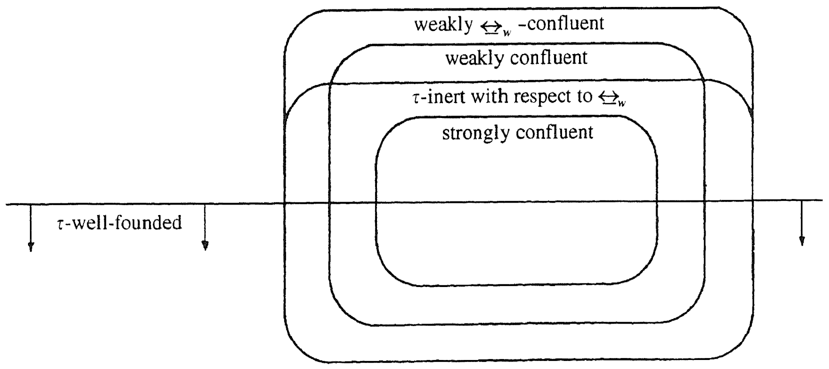

weakly $\leftrightarrow_w$ -confluent
weakly confluent
τ-inert with respect to $\leftrightarrow_w$
strongly confluent
τ-well-founded

Fig. 1. The transition systems in a diagram.

We conclude this section with an overview of the results, which have been depicted in Fig. 1. Below the horizontal line we have the τ-well-founded transition systems. We see that strong confluence always implies τ-inertness with respect to $\leftrightarrow_w$. The other two notions of confluence do not imply τ-inertness with respect to $\leftrightarrow_w$. However, the counterexamples are all τ-non-well-founded (above the horizontal line). Finally, we see that weak $\leftrightarrow_w$-confluence and τ-inertness with respect to $\leftrightarrow_w$ coincide for τ-well-founded transition systems (below the horizontal line).

4. Other graph equivalences

In this section we study the consequences of replacing $\leftrightarrow_w$ in Section 3 by other process equivalences. From the large variety of equivalences, that have been proposed to capture the behavioural aspects of processes (see e.g. [20]), we choose *strong bisimulation, branching bisimulation* and *finite trace equivalence*. This choice is motivated by the fact that these three equivalences are frequently used in the field of process theory.

4.1. Two definitions generalised

We generalise those notions of Section 3 that assume an equivalence relation on S.

Definition 4.1. Let $\sim\ \subseteq S \times S$ be an equivalence relation. A transition system $(S, \longrightarrow)$ is called a-inert with respect to $\sim$ iff

$$s \xrightarrow{a} t \Longrightarrow s \sim t \quad \text{for all } s,t \in S$$

Definition 4.2. A transition system $(S, \longrightarrow)$ is called weakly $\sim$-confluent for a iff for each pair $s \xrightarrow{a} t$ and $s \xrightarrow{\tau} s'$ of different steps there exist states u and u' such that

$u \sim u'$, $s' \overset{a}{\Rrightarrow} u'$ and $t \xrightarrow{\tau^*} u$. In a diagram:

$$\begin{tikzcd}
s \arrow[r, "\tau"] \arrow[d, "a"'] & s' \arrow[d, Rrightarrow, "a"'] \\
t \arrow[r, dashed, "\tau^*"] & u \sim u'
\end{tikzcd}$$

A transition system $(S, \longrightarrow)$ is called weakly $\sim$-confluent iff it is weakly $\sim$-confluent for a, for all $a \in \textsc{Act}$.

4.2. Strong bisimulation

About the state of affairs, in case we replace $\underline{\leftrightarrow}_w$ in Section 3 by $\underline{\leftrightarrow}$ (*strong bisimulation*[2]), we can be short. None of the notions of confluence, presented in this section, is sufficiently strong to imply τ-inertness with respect to $\underline{\leftrightarrow}$. This follows immediately from the (trivial) fact that even a strongly confluent transition system is not necessarily τ-inert with respect to $\underline{\leftrightarrow}$, e.g.

$$\begin{tikzcd}
s \arrow[r, "\tau"] & t \arrow[r, "a"] & u
\end{tikzcd}$$

with $a \not\equiv \tau$, is strongly confluent but not τ-inert with respect to $\underline{\leftrightarrow}$.

4.3. Branching bisimulation

In the previous subsection we explained that none of the notions of confluence implies τ-inertness with respect to $\underline{\leftrightarrow}$. Even stronger graph equivalences than $\underline{\leftrightarrow}$ of course give the same result and are therefore not interesting for us to analyse here.

Branching bisimulation, which we study in this subsection, however, is weaker than strong bisimulation (and stronger than weak bisimulation).

Definition 4.3. A relation $R \subseteq S \times S'$ is called a branching bisimulation on $(S, \longrightarrow)$ and $(S', \longrightarrow\!\!\!\blacktriangleright)$ iff

$$sRs' \Longrightarrow \begin{cases} s \xrightarrow{a} t \Longrightarrow [a \equiv \tau \ \wedge tRs'] \vee \\ \qquad [\exists u, u'.\, s' \xrightarrow{\tau^*}\!\!\blacktriangleright u \xrightarrow{a}\!\!\blacktriangleright u' \ \wedge \ sRu \ \wedge \ tRu'] \\ s' \xrightarrow{a}\!\!\blacktriangleright t' \Longrightarrow [a \equiv \tau \ \wedge sRt'] \vee \\ \qquad [\exists u, u'.\, s \xrightarrow{\tau^*} u \xrightarrow{a} u' \ \wedge \ s'Ru \ \wedge \ t'Ru'] \end{cases}$$

for all $s \in S$ and for all $s' \in S'$.

We say that R is a branching bisimulation on $(S, \longrightarrow)$ iff R is a weak bisimulation on $(S, \longrightarrow)$ and $(S, \longrightarrow)$. The union of all branching bisimulations is denoted as $\underline{\leftrightarrow}_b$.

[2] The definition for strong bisimulation can immediately be obtained from Definition 2.4 by replacing all triple arrows by single ones.

We provide the same theorems as for the weak bisimulation case. The proofs are analogous to the corresponding weak bisimulation versions of those theorems. Only a number of extra conditions must be checked. We omit those proofs.

Theorem 4.4. *Strongly confluent transition systems are τ-inert with respect to $\leftrightarrows_b$.*

Theorem 4.5. *Weakly confluent, τ-well-founded transition systems are τ-inert with respect to $\leftrightarrows_b$.*

Theorem 4.6. *Let $(S, \longrightarrow\!\triangleright)$ be τ-well-founded, then $(S, \longrightarrow\!\triangleright)$ is weakly $\leftrightarrows_b$-confluent iff $(S, \longrightarrow\!\triangleright)$ is τ-inert with respect to $\leftrightarrows_b$.*

4.4. Finite trace equivalence

In this subsection we summarize the consequences of replacing $\leftrightarrows_w$ in Section 3 by *finite trace equivalence*, which is denoted as $\approx$ in this paper. A more elaborate presentation of the results of this subsection can be found in [11].

Let $\textsc{Traces}(s)$ denote the set of finite traces starting in s.

Definition 4.7. Let $s, s' \in S$, then $s \approx s'$ iff $\textsc{Traces}(s) = \textsc{Traces}(s')$.

It is a well-known fact that $\leftrightarrows_w \subseteq \approx$ so statements like

weakly/strongly-confluent systems are τ-inert with respect to $\approx$

are trivial consequences of Theorems 3.2 and 3.5. However, a $\approx$-version of Theorem 3.9 (from left to right) is not trivially implied by Theorem 3.9, since the equivalence $\approx$ also appears in the notion of confluence. So not only the proof obligation but also the premise is weakened.

Theorem 4.8. *τ-well-founded, weakly $\approx$-confluent transition systems are τ-inert with respect to $\approx$.*

Proof. See [11]. □

5. Transition systems that are not τ-well-founded

Most of the results in Sections 3 and 4 rely on τ-well-foundedness of the transition system in question. However, many realistic examples of protocol specifications correspond to transition systems that are not τ-well-founded. As soon as a protocol internally consists of some kind of correction mechanism (e.g. retransmissions in a data link protocol) the specification of that protocol will contain a τ-loop. In Section 8.2 we see an example of this phenomenon.

Since we feel applicability to realistic examples is important, we considered the requirement that the transition system has to be τ-well-founded a serious drawback.

Therefore, we distinguish what we will call *progressing* τ-steps and *non-progressing* τ-steps. This enables us to formulate a slightly more subtle notion of confluence, which is sufficiently strong for our purposes and only relies on well-foundedness of the *progressing* τ-steps.

5.1. Progressing and non-progressing τ-steps

Convention 5.1. *We use the following notations:*

(i) $s \xrightarrow{\tau_>} t$ *for a progressing τ-step from s to t,*
(ii) $s \xrightarrow{\tau_<} t$ *for a non-progressing τ-step from s to t,*
(iii) $s \xrightarrow{\tau} t$ *for* $s \xrightarrow{\tau_>} t$ *or* $s \xrightarrow{\tau_<} t$,
(iv) $s \xRightarrow{a} t$ *for* $s \xrightarrow{\tau_>^*} s' \xrightarrow{a} t' \xrightarrow{\tau_>^*} t$,
(v) $s \overset{a}{\Rrightarrow} t$ *for* $s \xRightarrow{a} t \vee (a \equiv \tau \wedge s \xrightarrow{\tau_>^*} t)$.

Transition systems where the τ-steps are labelled with $>$ or with $<$ are called *τ-labelled transition systems*. Instead of τ-inertness with respect to $\sim$, we try to prove $\tau_>$-inertness with respect to $\sim$. In a formula:

$$s \xrightarrow{\tau_>} t \Longrightarrow s \sim t \tag{5}$$

Definition 2.4 of “weak bisimulation” remains unchanged for τ-labelled transition systems. Combined with Convention 5.1(iii) this means that the τ-steps mentioned in Definition 2.4 may either be progressing or not.

5.2. Progressing confluence

For each notion of confluence introduced in Section 3, as well as each notion introduced in Section 4, we can define a progressing version. Those progressing versions are given below in Definitions 5.2–5.4. As we will see, only the first two notions (Definitions 5.2 and 5.3) are useful to us. The notion of confluence, defined in Definition 5.4, does not imply $\tau_>$-inertness and is therefore not interesting. A counterexample is given in Proposition 5.9.

Definition 5.2. A τ-labelled transition system $(S, \longrightarrow)$ is called strongly $>$-confluent (pronounce: strongly progressing confluent) iff for each pair $s \xrightarrow{\tau_>} s'$ and $s \xrightarrow{a} t$ of different steps there exists a state t' such that $t \xrightarrow{\tau_>} t'$ and $s' \xrightarrow{a} t'$. In a diagram:

$$\begin{tikzcd} s \arrow[r, "\tau_>"] \arrow[d, "a"'] & s' \arrow[d, dashed, "a"] \\ t \arrow[r, dashed, "\tau_>"] & t' \end{tikzcd}$$

Definition 5.3. A τ-labelled transition system $(S, \longrightarrow)$ is called weakly $>$-confluent (pronounce: weakly progressing confluent) iff for each pair $s \xrightarrow{\tau_>} s'$ and $s \xrightarrow{a} t$ of different steps there exists a state t' such that $t \xrightarrow{\tau_>^*} t'$ and $s' \xRightarrow{a} t'$. In a diagram:

$$\begin{tikzcd}
s \arrow[r, "\tau_>"] \arrow[d, "a"'] & s' \arrow[d, Rightarrow, dashed, "a"'] \\
t \arrow[r, dashed, "\tau_>^*"] & t'
\end{tikzcd}$$

Definition 5.4. A τ-labelled transition system $(S, \longrightarrow)$ is called weakly $>$-$\sim$-confluent (pronounce: weakly progressing $\sim$-confluent) iff for each pair $s \xrightarrow{\tau_>} s'$ and $s \xrightarrow{a} t$ of different steps there exist states u and u' such that $t \xrightarrow{\tau_>^*} u$, $s' \xRightarrow{a} u'$ and $u \sim u'$. In a diagram:

$$\begin{tikzcd}
s \arrow[rr, "\tau_>"] \arrow[d, "a"'] & & s' \arrow[d, Rightarrow, dashed, "a"'] \\
t \arrow[r, dashed, "\tau_>^*"] & u \arrow[r, phantom, "\sim"] & u'
\end{tikzcd}$$

Theorems 3.2 and 3.5 have a progressing counterpart, stating that the progressing version of the confluence notion in question implies $\tau_>$-inertness with respect to $\sim$.

Theorem 5.5. *Strongly $>$-confluent transition systems are $\tau_>$-inert with respect to $\sim$, for $\sim \in \{\leftrightarrows_w, \leftrightarrows_b, \approx\}$.*

Proof. Analogous to the corresponding proofs in Sections 3 and 4. $\square$

Lemma 5.6. *Let $(S, \longrightarrow)$ be $\tau_>$-well-founded and weakly $>$-confluent for $\tau_>$. Let $s \xrightarrow{\tau_>^*} s'$ and $s \xrightarrow{\tau_>^*} t$, then there exists a t' such that $s' \xrightarrow{\tau_>^*} t'$ and $t \xrightarrow{\tau_>^*} t'$. In a diagram:*

$$\begin{tikzcd}
s \arrow[r, "\tau_>^*"] \arrow[d, "\tau_>^*"'] & s' \arrow[d, dashed, "\tau_>^*"'] \\
t \arrow[r, dashed, "\tau_>^*"] & t'
\end{tikzcd}$$

Proof. Analogously to the proof of Lemma 3.4. $\square$

Lemma 5.7. *Let* $(S, \longrightarrow)$ *be* $\tau_>$*-well-founded and weakly* $>$*-confluent, then*

$$\mathcal{T}_> = \{(x,y) \mid x \xrightarrow{\tau_>^*} y\}$$

is a weak bisimulation.

Proof. Let $s\mathcal{T}_>^* s'$, i.e. $s \xrightarrow{\tau_>^*} s'$, then we have to show:

(i) $s \xrightarrow{a} t \Longrightarrow \exists t'.\ s' \overset{a}{\Longrightarrow} t' \wedge t\mathcal{T}_>^* t'$

(ii) $s' \xrightarrow{a} t' \Longrightarrow \exists t.\ s \overset{a}{\Longrightarrow} t \wedge t\mathcal{T}_>^* t'$

(ii) is easy, take $t \equiv t'$ then $s \overset{a}{\Longrightarrow} t$ since $s \xrightarrow{\tau^*} s' \xrightarrow{a} t'$. Furthermore, $t\mathcal{T}_>^* t'$ holds by reflexivity of $\mathcal{T}_>^*$.

By induction to $d_{\tau_>}(s)$ we show that (i) holds. If $s \equiv s'$ (and, in particular, if $d_{\tau_>}(s) = 0$) the lemma is trivial: take $t' \equiv t$. Assume that (i) holds for all states x with $d_{\tau_>}(x) < d_{\tau_>}(s)$. Let $s \xrightarrow{\tau_>} s'' \xrightarrow{\tau_>^*} s'$. We are in the following situation:

$$\begin{tikzcd}
s \arrow[r, "\tau_>"] \arrow[d, "a"'] & s'' \arrow[r, "\tau_>^*"] & s' \\
t & &
\end{tikzcd}$$

In case $a \equiv \tau_>$ then we apply Lemma 5.6. If $a \equiv \tau_<$ then there exists (by weak $>$-confluence) a state t' such that $t \xrightarrow{\tau_>^*} t'$ and $s' \overset{\tau_<}{\Longrightarrow} t'$. If $s' \xrightarrow{\tau_>^*} t'$ then we can draw the following diagram:

$$\begin{tikzcd}
s \arrow[r, "\tau_>"] \arrow[d, "\tau_<"'] & s'' \arrow[r, "\tau_>^*"] \arrow[d, "\tau_>^*"'] & s' \arrow[d, "\tau_>^*"'] \\
t \arrow[r, "\tau_>^*"] & t'' \arrow[r, "\tau_>^*"] & t'
\end{tikzcd}$$

The right part of the diagram is given by Lemma 5.6.

The case $s' \xrightarrow{\tau_>^*} u \xrightarrow{\tau_<} u' \xrightarrow{\tau_>^*} t'$ is treated like the general case $a \not\equiv \tau$.

If $a \not\equiv \tau$ then we can draw the diagram as in Fig. 2. The left part of the diagram is given by weak $>$-confluence. The upper-right part is given by Lemma 5.6. The middle-right part is given by applying the induction hypothesis on $u\mathcal{T}_>^* v$ and $u \xrightarrow{a} u'$, using that $d_{\tau_>}(u) < d_{\tau_>}(s)$. Finally, the lower-right part of the diagram is given by applying Lemma 5.6 again. We are done because $t\mathcal{T}_>^* t'$ and $s' \overset{a}{\Longrightarrow} t'$. □

Theorem 5.8. *Weakly* $>$*-confluent,* $\tau_>$*-well-founded transition systems are* $\tau_>$*-inert with respect to* $\sim$*, for* $\sim \in \{\leftrightarroweq_w, \leftrightarroweq_b, \approx\}$.

Proof. $\mathcal{T}_> \subseteq \leftrightarroweq_w$ by Lemma 5.7. Now $s \xrightarrow{\tau_>} t \Longrightarrow s\mathcal{T}_> t \Longrightarrow s \leftrightarroweq_w t$, which proves the theorem for $\sim \equiv \leftrightarroweq_w$ (and hence also for $\sim \equiv \approx$). In order to prove the theorem for $\sim \equiv \leftrightarroweq_b$ some extra conditions have to be checked. □

$$\begin{tikzcd}
s \arrow[r, "\tau_>"] \arrow[ddd, "a"'] & s'' \arrow[r, "\tau_>^*"] \arrow[d, "\tau_>^*"'] & s'' \arrow[d, "\tau_>^*"] \\
 & u \arrow[r, "\tau_>^*"] \arrow[d, "a"'] & v \arrow[d, Rightarrow, "a"] \\
 & u' \arrow[r, "\tau_>^*"] \arrow[d, "\tau_>^*"'] & v' \arrow[d, "\tau_>^*"] \\
t \arrow[r, "\tau_>^*"] & t'' \arrow[r, "\tau^*"] & t'
\end{tikzcd}$$

Fig. 2.

Proposition 5.9. *Let $(S, \longrightarrow\!\!\triangleright)$ be weakly $<$-$\sim$-confluent and $\tau_>$-well-founded, then $(S, \longrightarrow\!\!\triangleright)$ is not necessarily $\tau_>$-inert with respect to $\sim$, for $\sim \in \{\underline{\leftrightarrow}_w, \underline{\leftrightarrow}_b, \approx\}$.*

Proof. Let $a \not\equiv \tau$. The following transition system is not $\tau_>$-inert with respect to $\sim$ since $s_4 \xrightarrow{\tau_>} s_6$ and $s_4 \not\sim s_6$. However, weak $<$-$\sim$-confluence holds.

$$\begin{tikzcd}
s_1 & s_2 \arrow[l, "\tau_<"'] \arrow[d, "a"'] & s_3 \arrow[l, "\tau"'] \arrow[r, shift left, "\tau_>"] & s_4 \arrow[l, shift left, "\tau_<"] \arrow[d, "\tau_>"] \\
 & s_5 & & s_6
\end{tikzcd}$$

Suppose $R \subseteq S \times S$ is defined by $s_1 R s_5 R s_6$ and $s_2 R s_3 R s_4$, then one easily verifies that $R^{\circledast}$ is a branching bisimulation on S. Now $(S, \longrightarrow\!\!\triangleright)$ is weakly $>$-$\underline{\leftrightarrow}_b$-confluent because

$$\begin{tikzcd}
s_3 \arrow[r, "\tau_>"] \arrow[d, "\tau"'] & s_4 \arrow[d, "\tau_>^*"] \\
s_2 \arrow[r, "\tau_>^*"] & s_2 \underline{\leftrightarrow}_b s_4
\end{tikzcd}
\quad \text{and} \quad
\begin{tikzcd}
s_4 \arrow[r, "\tau_>"] \arrow[d, "\tau_<"'] & s_6 \arrow[d, "\tau_<^*"] \\
s_3 \arrow[r, "\tau_>^*"] & s_6 \underline{\leftrightarrow}_b s_6
\end{tikzcd}$$

Since $\underline{\leftrightarrow}_b \subseteq \underline{\leftrightarrow}_w \subseteq \approx$ we know that the transition system is also weakly $>$-$\underline{\leftrightarrow}_w$-confluent and weakly $>$-$\approx$-confluent so we are done. □

So far, we showed that weak $>$-confluence is useful to us and weak $>$-$\sim$-confluence is not. One might wonder whether there are other ways to relax the notion of weak $>$-confluence. The obvious way to do this is to allow non-progressing τ-steps in either the path $t \xrightarrow{\tau} t'$ or the path $s' \xRightarrow{a} t'$. We show that the notions of confluence, thus obtained (weak $>$-confluence$_1$ and weak $>$-confluence$_2$), both do not imply $\tau_>$-inertness and are therefore not useful to us.

Definition 5.10. A system $(S, \longrightarrow)$ is called weakly $>$-confluent$_1$ iff for each pair $s \xrightarrow{a} t$ and $s \xrightarrow{\tau_>} s'$ of different steps there exists a state t' such that $t \xrightarrow{\tau^*} t'$ and $s' \xRightarrow{a} t'$. In a diagram:

$$\begin{tikzcd}
s \arrow[r, "\tau_>"] \arrow[d, "a"'] & s' \arrow[d, Rightarrow, "a"'] \\
t \arrow[r, dashed, "\tau^*"] & t'
\end{tikzcd}$$

Now the following transition system is weakly $>$-confluent$_1$ but not $\tau_>$-inert with respect to $\sim$, for $\sim \in \{\underline{\leftrightarrow}_w, \underline{\leftrightarrow}_b, \approx\}$.

$$\begin{tikzcd}
s \arrow[r, "a"] \arrow[d, "\tau_>"'] & s' \arrow[r, "b"] \arrow[d, "\tau_<"'] & s'' \\
t \arrow[r, "a"] & t' &
\end{tikzcd} \qquad (6)$$

$s \not\underline{\leftrightarrow}_w t$ although they are connected by a progressing τ-step. It is essential in this example that the τ-step from s' to t' is non-progressing. It relieves us from the obligation to add a state t'' satisfying $t' \xRightarrow{b} t''$ and $s'' \xrightarrow{\tau^*} t''$. Note that (6) is not weakly $>$-confluent since the τ-step from s' to t' is non-progressing.

Definition 5.11. A system $(S, \longrightarrow)$ is called weakly $>$-confluent$_2$ iff for each pair $s \xrightarrow{a} t$ and $s \xrightarrow{\tau_>} s'$ of different steps there exists a state t' such that $t \xrightarrow{\tau_>^*} t'$ and $s' \xRightarrow{a} t'$. In a diagram:

$$\begin{tikzcd}
s \arrow[r, "\tau_>"] \arrow[d, "a"'] & s' \arrow[d, Rightarrow, "a"'] \\
t \arrow[r, dashed, "\tau_>^*"] & t'
\end{tikzcd}$$

The following transition system is weakly $>$-confluent$_2$ but not $\tau_>$-inert with respect to $\sim$, for $\sim\in\{\underline{\leftrightarrow}_w,\underline{\leftrightarrow}_b,\approx\}$.

$$\begin{tikzcd}
s \arrow[rr, shift left, "\tau_>"] \arrow[d, "a"'] & & s' \arrow[ll, shift left, "\tau_<"] \arrow[d, "\tau_>"'] \\
t & & t'
\end{tikzcd} \tag{7}$$

The progressing τ-step from s' to t' does not connect weakly bisimilar states. Diagram (7) is weakly $>$-confluent$_2$ because the following properties hold:

$$\begin{tikzcd}
s \arrow[r, "\tau_>"] \arrow[d, "a"'] & s' \arrow[d, Rightarrow, "a"'] \\
t \arrow[r, "\tau_>^*"] & t
\end{tikzcd}
\qquad
\begin{tikzcd}
s' \arrow[r, "\tau_>"] \arrow[d, "\tau_<"'] & t' \arrow[d, Rightarrow, "\tau_<"'] \\
s \arrow[r, "\tau_>^*"] & t'
\end{tikzcd}$$

6. Confluence of linear processes

We want to use the notion confluence to verify the correctness of processes. In order to do so, we must be able to determine whether a transition system is confluent. This is in general not possible, because the transition systems belonging to distributed systems are often too large to be handled as plain objects. In order to manipulate with large state spaces (*Clustered*) *Linear Process* ((C-)LPs) [7] can be used as in these C-LPs the state space is compactly encoded using data parameters. Moreover, processes that are described using the common process algebra operators, including parallelism, can straightforwardly be transformed to a C-LP, maintaining strong bisimulation.

In this section we describe how a C-LP can be shown to be confluent. In the next section we show how confluence is used to reduce the size of state spaces.

6.1. Linear processes

Definition 6.1. Let $Act\subseteq \textsc{Act}$ be a finite set of actions, containing a special action δ representing deadlock or inaction (see [5,4]). A clustered[3] linear process equation is an expression of the form

$$p(d)=\sum_{a\in Act}\sum_{e_a:E_a} a(f_a(d,e_a))\cdot p(g_a(d,e_a))\lhd b_a(d,e_a)\rhd\delta$$

[3] In earlier works these processes are sometimes called "deterministic", referring to the fact that the summand is determined by the action. This name is unfortunate however, because determinism of the process in question is usually not the case.

for data sorts D, E_a and F_a and functions $f_a : D \times E_a \longrightarrow F_a$, $g_a : D \times E_a \longrightarrow D$ and $b_a : D \times E_a \longrightarrow \mathbb{B}$ where $\mathbb{B}$ is the predefined sort of booleans. We assume that the internal action τ ($\tau_>$ and $\tau_<$ if progressing and non-progressing τ's are distinguished) has no data parameter.

In [7] summands without a recursive call are also allowed in the definition of a linear process. We omit these summands here.

It is straightforward to see how a linear process equation determines a transition system. The process $p(d)$ can perform an action $a(f_a(d,e_a))$ for every $a \in Act$ and every data element e_a of sort E_a, provided the condition $b_a(d,e_a)$ holds. The process then continues as $p(g_a(d,e_a))$. Hence, the notions defined in the previous sections carry over directly.

A linear process is called *convergent* iff the corresponding transition system is τ-well-founded. If we distinguish progressing and non-progressing τ's, we use the notion convergence with respect to the progressing τ's (i.e. $\tau_>$).

Definition 6.2. A linear process as defined in Definition 6.1 is called $>$*-convergent* iff there is a well-founded ordering $<$ on D such that for all $d : D$ and $e_{\tau_>} : D_{\tau_>}$ if $b_{\tau_>}(d, e_{\tau_>})$, then $g_{\tau_>}(d, e_{\tau_>}) < d$.

The $>$ symbol in "$>$-convergent" refers to "progressing" and not to the ordering on D. However, the ordering on D and the labelling on τ-steps are closely related. Typically, the τ-steps that are labelled with $>$ are precisely those τ-steps $p(d) \xrightarrow{\tau} p(d')$ satisfying $d' < d$. So after each progressing τ-step one is moved towards a state with a value that is strictly smaller with respect to some well-founded ordering. Thus, the progressing τ-steps express progression in the sense that progression is made in the execution of internal activity.

6.2. *A condition for strong confluence*

We provide sufficient criteria for p to be strongly confluent. Let p be a clustered linear process as defined in Definition 6.1. The criteria can best be understood via the following diagram.

$$\begin{tikzcd}
p(d) \arrow[r, "\tau"] \arrow[d, "a(f_a(d,e_a))"'] & p(g_\tau(d,e_\tau)) \arrow[d, dashed, "a(f_a(g_\tau(d,e_\tau),e'_a))"'] \\
p(g_a(d,e_a)) \arrow[r, dashed, "\tau"] & \begin{array}{l} p(g_\tau(g_a(d,e_a),e'_\tau)) = \\ p(g_a(g_\tau(d,e_\tau),e'_a)) \end{array}
\end{tikzcd}$$

Note that in this diagram $p(g_a(d,e_a))$ and $p(g_\tau(d,e_\tau))$ are supposed to be different if $a = \tau$. We summarise the conditions in the following theorem.

Theorem 6.3. *The process p as defined in Definition* 6.1 *is strongly confluent if for all $a \in Act$, $e_1 : E_a$, $e_2 : E_\tau$ such that*

(i) $a = \tau \Rightarrow g_a(d,e_1) \neq g_\tau(d,e_2)$

(ii) $b_a(d,e_1) \wedge b_\tau(d,e_2)$

the following property holds:

$$\exists e_3 : E_a,\ e_4 : E_\tau \begin{cases} f_a(d,e_1) = f_a(g_\tau(d,e_2),e_3) \wedge \\ b_a(g_\tau(d,e_2),e_3) \wedge \\ b_\tau(g_a(d,e_1),e_4) \wedge \\ g_a(g_\tau(d,e_2),e_3) = g_\tau(g_a(d,e_1),e_4). \end{cases}$$

6.3. A condition for weak progressing confluence

In this section we derive a condition to establish that a C-LP is weakly confluent. This is more involved, because we must now speak about sequences of transitions.

In order to keep the notation compact, we introduce some convenient abbreviations. Let $\sigma, \sigma', \ldots$ range over lists of pairs $\langle a, e_a \rangle$ with $a \in Act$ and $e_a : E_a$. We define $\mathcal{G}_d(\sigma)$ with $d \in D$ by induction over the length of σ:

$$\mathcal{G}_d(\lambda) = d \qquad \mathcal{G}_d(\sigma\langle a, e_a\rangle) = g_a(\mathcal{G}_d(\sigma), e_a)$$

Each σ determines an execution fragment:

$$\underbrace{p(d) \xrightarrow{\qquad\qquad} p(\mathcal{G}_d(\sigma))}_{\text{determined by } \sigma} \xrightarrow{a(f_a(\mathcal{G}_d(\sigma),e_a))} p(\mathcal{G}_d(\sigma\langle a, e_a\rangle))$$

is the execution fragment determined by $\sigma\langle a, e_a\rangle$. This execution fragment is allowed for $p(d)$ iff the conjunction $\mathcal{B}_d(\sigma)$ of all conditions associated to the actions in σ evaluates to $\top$. The boolean $\mathcal{B}_d(\sigma)$ is also defined by induction to the length of σ:

$$\mathcal{B}_d(\lambda) = \top \qquad \mathcal{B}_d(\sigma\langle a, e_a\rangle) = \mathcal{B}_d(\sigma) \wedge b_a(\mathcal{G}_d(\sigma), e_a)$$

We write $\pi_1(\sigma)$ for the sequence of actions that is obtained from σ by applying the first projection to all its elements.

$$\pi_1(\lambda) = \lambda$$
$$\pi_1(\sigma\langle a, e_a\rangle) = \pi_1(\sigma)a$$

The diagram for weak $>$-confluence can be redrawn instantiated for C-LPs as shown below. The actions in the (possibly empty) sequences σ_1, σ_2 and σ_3 must all be progressing τ-steps, that is $\pi_1(\sigma_i) = \tau^*_>$ for all $i = 1,2,3$.

$$\begin{tikzcd}
p(d) \arrow[r, "\tau_>"] \arrow[d, "a(f_a(d,e_a))"'] & p(g_{\tau_>}(d,e_{\tau_>})) \arrow[d, Rightarrow, dashed, "a(f_a(\mathcal{G}_{g_{\tau_>}(d,e_{\tau_>})}(\sigma_1),e'_a))"'] \\
p(g_a(d,e_a)) \arrow[r, dashed, "\tau^*_>"] & \begin{array}{l} p(\mathcal{G}_{g_a(d,e_a)}(\sigma_3) = \\ p(\mathcal{G}_{g_{\tau_>}(d,e_{\tau_>})}(\sigma_1\langle a,e'_a\rangle\sigma_2)) \end{array}
\end{tikzcd}$$

We summarise this diagram in the following theorem. Due to its generality the theorem looks rather complex. However, in those applications that we considered, the lists that are existentially quantified were mainly empty, which trivialises major parts of the theorem.

Theorem 6.4. *The process p as defined in Definition 6.1 is weakly $>$-confluent if p is $>$-convergent and for all $e_1{:}E_a$, $e_2{:}E_{\tau_>}$ such that*

(i) $a = \tau_> \Rightarrow g_a(d,e_1) \neq g_{\tau_>}(d,e_2)$

(ii) $b_a(d,e_1) \wedge b_{\tau_>}(d,e_2)$

the following property holds:

$$\exists \sigma_1, e_3, \sigma_2, \sigma_3 \begin{cases} \pi_1(\sigma_i) = \tau^*_> \text{ for all } i = 1,2,3 \wedge \\ f_a(d,e_1) = f_a(\mathcal{G}_{g_{\tau_>}(d,e_2)}(\sigma_1),e_3) \wedge \\ \mathcal{B}_{g_a(d,e_1)}(\sigma_3) \wedge \\ \mathcal{B}_{g_{\tau_>}(d,e_2)}(\sigma) \wedge \\ \mathcal{G}_{g_a(d,e_1)}(\sigma_3) = \mathcal{G}_{g_{\tau_>}(d,e_2)}(\sigma) \end{cases}$$

where $\sigma = \sigma_1\langle a,e_3\rangle\sigma_2$, or $a = \tau$ and $\sigma = \sigma_1\sigma_2$.

7. State space reduction

Here we employ the results about confluence and τ-inertness that we have obtained thus far to achieve state space reductions and to simplify the behaviour of processes. In this section we work in the setting of branching bisimulation. Contrary to the situation in Section 4 this immediately implies that the results apply to weak bisimulation as well. First we present the results on transition systems in general, and then on linear processes. This is done because for transition systems the results are easier to understand. However, as argued in the previous section, the results can be applied more conveniently in the setting of linear processes.

Definition 7.1. Let $T_1 = (S, \longrightarrow\!\!\triangleright)$ and $T_2 = (S, \longrightarrow\!\!\blacktriangleright)$ be τ-labelled transition systems. We call T_2 a τ-Prioritised-reduction (TP-reduction) of T_1 iff

(i) $\longrightarrow\!\!\blacktriangleright \subseteq \longrightarrow\!\!\triangleright$,

(ii) for all $s,s' \in S$ if $s \xrightarrow{a}\!\!\triangleright s'$ then $s \xrightarrow{a}\!\!\blacktriangleright s'$ or $s \xrightarrow{\tau_>}\!\!\blacktriangleright s''$ for some s''.

Clearly, T_2 can be obtained from T_1 by iteratively removing transitions from states as long as these keep at least one outgoing progressing τ-step. It goes without saying that this may considerably reduce the state space of T_1, especially because large parts may become unreachable.

The following theorem states that if T_1 is $\tau_>$-inert with respect to $\leftrightarrow_b$, then a TP-reduction maintains weak bisimulation. As confluence implies τ-inertness, this theorem explains how confluence can be used to reduce the size of transition systems.

Theorem 7.2. *Let $T_1 = (S, \longrightarrow\!\!\triangleright)$ and $T_2 = (S, \longrightarrow\!\!\blacktriangleright)$ be τ-labelled transition systems. Let $\leftrightarrow_b$ be the maximal branching bisimulation on T_1 and T_2. If T_1 is $\tau_>$-inert with respect to $\leftrightarrow_b$ and T_2 is a τ-well founded TP-reduction of T_1 then $s \leftrightarrow_b s$ for each state $s \in S$.*

Proof. Let R denote the union of all branching bisimulations on T_1. Since T_1 is $\tau_>$-inert with respect to $\leftrightarrow_b$ we have (by definition) that

$$s \xrightarrow{\tau_>}\!\!\triangleright\, s' \implies sRs' \tag{8}$$

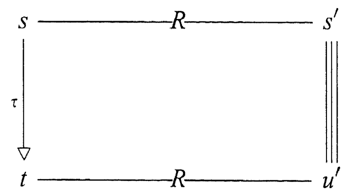
holds. We prove that R is a branching bisimulation on T_1 and T_2. Then, by definition, $R \subseteq \leftrightarrow_b$ and the theorem follows immediately since sRs for all $s \in S$.

Let sRs'. We have to prove:

(i) $s \xrightarrow{a}\!\!\triangleright\, t \implies [a \equiv \tau \wedge tRs'] \vee [\exists u, u'. s' \xrightarrow{\tau^*}\!\!\blacktriangleright\, u \xrightarrow{a}\!\!\blacktriangleright\, u' \wedge sRu \wedge tRu']$

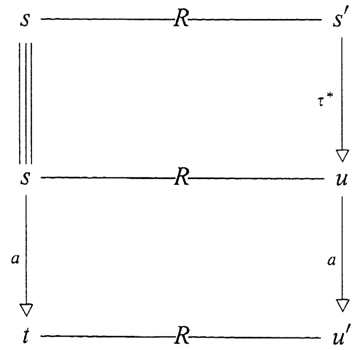
(ii) $s' \xrightarrow{a}\!\!\blacktriangleright\, t' \implies [a \equiv \tau \wedge sRt'] \vee [\exists u, u'. s \xrightarrow{\tau^*}\!\!\triangleright\, u \xrightarrow{a}\!\!\triangleright\, u' \wedge s'Ru \wedge t'Ru']$

We first prove (i). Suppose $s \xrightarrow{a}\!\!\triangleright\, t$. We are in one of the following situations:



$$\begin{tikzcd} s \arrow[r, dash, "R"] \arrow[d, "\tau"'] & s' \arrow[d, equal] \\ t \arrow[r, dash, "R"] & u' \end{tikzcd}$$

or



$$\begin{tikzcd} s \arrow[r, dash, "R"] \arrow[d, equal] & s' \arrow[d, "\tau^*"] \\ s \arrow[r, dash, "R"] \arrow[d, "a"'] & u \arrow[d, "a"] \\ t \arrow[r, dash, "R"] & u' \end{tikzcd}$$

The existence of u and u' as depicted above is given by the definition of R. If all transitions of $s' \longrightarrow\!\!\triangleright\!\!\triangleright\, u'$ occur in T_2 (and in particular if $s' \equiv u'$) then we are done. To settle (i) in the other case – i.e. the case that not all transitions of $s' \longrightarrow\!\!\triangleright\!\!\triangleright\, u'$ occur in T_2 (and in particular $s' \not\equiv u'$) – we prove that the following property holds:

$$\begin{tikzcd}
s \arrow[r, dash, "R"] \arrow[d, equal] & s' \arrow[d, "\tau^*", -{Triangle[open]}] & & s \arrow[r, dash, "R"] \arrow[d, equal] & s' \arrow[d, dashed, "\tau^*", -{Triangle}] \\
s \arrow[r, dash, "R"] \arrow[d, "a"', -{Triangle[open]}] & u \arrow[d, "a", -{Triangle[open]}] & \Longrightarrow & s \arrow[r, dash, dashed, "R"] \arrow[d, "a"', -{Triangle[open]}] & y \arrow[d, dashed, "a", -{Triangle}] \\
t \arrow[r, dash, "R"] & u' & & t \arrow[r, dash, dashed, "R"] & y'
\end{tikzcd}$$

We use induction on $d_\tau(s')$. Here, d_τ is defined with respect to T_2, i.e. $d_\tau(x)$ equals the number of $\overset{\tau}{\longrightarrow\!\!\!\!\blacktriangleright}$-steps that can be executed from x.

Let v and v' be those states occurring in the path $s' \overset{\tau^*}{\longrightarrow\!\!\!\!\triangleright} u \overset{a}{\longrightarrow\!\!\!\!\triangleright} u'$ of T_1, such that $s' \overset{\tau^*}{\longrightarrow\!\!\!\!\blacktriangleright} v \not\longrightarrow\!\!\!\!\blacktriangleright v'$. That is, $v \longrightarrow\!\!\!\!\triangleright v'$ is the first transition of $s' \overset{\tau^*}{\longrightarrow\!\!\!\!\triangleright} u \overset{a}{\longrightarrow\!\!\!\!\triangleright} u'$ that does not occur in T_2. We can draw the diagram as in Fig. 3. If $s' \equiv v$ then $v \longrightarrow\!\!\!\!\triangleright v'$ is the first transition of $s' \overset{\tau^*}{\longrightarrow\!\!\!\!\triangleright} u \overset{a}{\longrightarrow\!\!\!\!\triangleright} u'$, if $v \equiv u$ then $v \longrightarrow\!\!\!\!\triangleright v'$ is the last transition of $s' \overset{\tau^*}{\longrightarrow\!\!\!\!\triangleright} u \overset{a}{\longrightarrow\!\!\!\!\triangleright} u'$ and otherwise $v \longrightarrow\!\!\!\!\triangleright v'$ is an intermediate transition of $s' \overset{\tau^*}{\longrightarrow\!\!\!\!\triangleright} u \overset{a}{\longrightarrow\!\!\!\!\triangleright} u'$.

The transition $v \overset{\tau_>}{\longrightarrow\!\!\!\!\blacktriangleright} z$ exists because T_2 is a TP-reduction of T_1. Since $\longrightarrow\!\!\!\!\blacktriangleright \subseteq \longrightarrow\!\!\!\!\triangleright$ we can conclude vRz, using (8). Furthermore, sRz (by transitivity of R) and $d_\tau(z) < d_\tau(s')$. Since R is a branching bisimulation on T_1 there exist states x and x' as depicted in Fig. 4. Now, by the induction hypothesis, there exist states y and y' as shown in Fig. 5, and we are done because y and y' satisfy the required properties.

$$\begin{tikzcd}
s \arrow[r, dash, "R"] \arrow[d, equal] & s' \arrow[d, "\tau^*", -{Triangle}] & \\
s \arrow[r, dash, "R"] \arrow[d, equal] & v \arrow[d, "\tau^*", -{Triangle[open]}] \arrow[r, "\tau_>", -{Triangle}] & z \\
s \arrow[r, dash, "R"] \arrow[d, "a"', -{Triangle[open]}] & u \arrow[d, "a", -{Triangle[open]}] & \\
t \arrow[r, dash, "R"] & u' &
\end{tikzcd}$$

Fig. 3.

$$\begin{tikzcd}
s \arrow[r, dash, "R"] \arrow[dd, equal] & v \arrow[r, "\tau_>"] \arrow[d, "\tau^*"'] & z \arrow[dd, "\tau^*"'] \\
 & u \arrow[d, dotted, no head] & \\
s \arrow[rr, dash, "R"] \arrow[d, "a"'] & & x \arrow[d, "a"'] \\
t \arrow[rr, dash, "R"] & & x'
\end{tikzcd}$$

Fig. 4.

$$\begin{tikzcd}
s \arrow[r, dash, "R"] \arrow[dd, equal] & v \arrow[r, "\tau_>"] \arrow[d, "\tau^*"'] & z \arrow[dd, "\tau^*"'] \\
 & u \arrow[d, dotted, no head] & \\
s \arrow[rr, dash, "R"] \arrow[d, "a"'] & & y \arrow[d, "a"'] \\
t \arrow[rr, dash, "R"] & & y'
\end{tikzcd}$$

Fig. 5.

The validity of (ii) is easy. Suppose $s' \xrightarrow{a} t'$. We are in the following situation:

$$\begin{tikzcd}
s \arrow[r, dash, "R"] & s' \arrow[d, "a"'] \\
 & t'
\end{tikzcd}$$

Since $\longrightarrow\!\!\blacktriangleright \subseteq \longrightarrow\!\!\triangleright$ we have that $s' \xrightarrow{\tau}\!\!\triangleright\, t'$. Now there exist states u and u' as depicted below since R is a branching bisimulation on T_1.

$$\begin{tikzcd}
s \arrow[r, dash, "R"] \arrow[d, "\tau^*"'] & s' \arrow[d, equal] \\
u \arrow[r, dash, "R"] \arrow[d, "a"'] & s' \arrow[d, "a"] \\
u' \arrow[r, dash, "R"] & t'
\end{tikzcd}$$

so we are done. $\square$

Below we reformulate the notion of a TP-reduction on linear processes. We assume that p is a linear process according to Definition 6.1 and that the data sort E_τ is ordered by some total ordering $\prec$, which assigns priority among τ-actions in the TP-reduction.

Definition 7.3. The TP-reduction of p is the linear process

$$p_r(d) = \sum_{a \in Act} \sum_{e_a : E_a} a(f_a(d,e_a))\, p_r(g_a(d,e_a)) \lhd b_a(d,e_a) \wedge c_a(d,e_a) \rhd \delta$$

where

$$c_a(d,e_a) \equiv \begin{cases} \neg\exists e_{\tau_>} : E_{\tau_>}\ b(d,e_{\tau_>}) & \text{if } a \neq \tau_> \\ \neg\exists e_{\tau_>} : E_{\tau_>}\ e_a \prec e_{\tau_>} \wedge b(d,e_{\tau_>}) & \text{if } a = \tau_> \end{cases}$$

Note that, for the sake of conciseness, we use $\exists$ in the condition $c_a(d,e_a)$, which actually does not adhere to the formal definition of μCRL.

Theorem 7.4. *If the linear process p is $>$-convergent and weakly $>$-confluent, and if p_r is convergent, then for all $d:D$*

$$\boxed{p(d) \leftrightarrows_b p_r(d)}$$

Once again we remark that the results of this section Theorems (7.2 and 7.4) also hold for weak bisimulation.

8. Two examples

We illustrate how we apply the theory by means of two examples, where the structure of the processes is considerably simplified by a confluence argument.

8.1. Concatenation of two queues

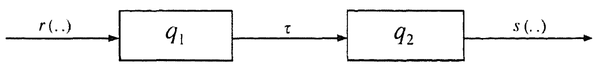
Consider the following linear process $Q(q)$ describing a queue q:

$$Q(q) = \sum_{e_r:E_r} r(e_r) \cdot Q(in(e_r,q)) + s(toe(q)) \cdot Q(untoe(q)) \lhd ne(q) \rhd \delta$$

The boolean expression $ne(q)$ evaluates to $\top$ iff q is not empty. The function *in* is used to insert an element to a queue and the function *untoe* is used to remove that element of a queue which has been inserted first. The function *toe* returns this first element. Now the following linear process $Q(\langle q_1,q_2\rangle)$ describes the concatenation of two queues q_1 and q_2:

$$\begin{aligned} Q(\langle q_1,q_2\rangle) = \textstyle\sum_{e_r:E_r} r(e_r) \cdot{}& Q(\langle in(e_r,q_1),q_2\rangle) && \lhd \ \top \ \rhd \delta + \\ \tau \cdot{}& Q(\langle untoe(q_1), in(toe(q_1),q_2)\rangle) && \lhd ne(q_1) \rhd \delta + \\ s(toe(q_2)) \cdot{}& Q(\langle q_1, untoe(q_2)\rangle) && \lhd ne(q_2) \rhd \delta \end{aligned}$$

As we can see, the process $Q(\langle q_1,q_2\rangle)$ can always read a datum and insert it in q_1. If q_2 is not empty then the “toe” of q_2 can be sent. The internal action τ removes the first element of q_1 and inserts it in q_2.



Using Theorem 6.3 we can straightforwardly prove that $Q(\langle q_1,q_2\rangle)$ is strongly confluent. For the read action r we find the condition that for all queues q_1,q_2 and $e_r:E_r$

$$ne(q_1) \Rightarrow \exists e_r':E_r \; e_r = e_r' \wedge ne(in(d,q_1)) \wedge$$

$$\langle in(e_r', untoe(q_1)), in(toe(q_1),q_2)\rangle = \langle untoe(in(e_r,q_1)), in(toe(in(e_r,q_1)),q_2)\rangle.$$

Similarly, we can formulate the following conditions for the action s. For all queues q_1,q_2

$$(ne(q_2) \wedge ne(q_1)) \Rightarrow$$

$$toe(q_2) = toe(in(toe(q_1),q_2)) \wedge$$

$$ne(in(toe(q_1),q_2)) \wedge ne(q_1) \wedge$$

$$\langle untoe(q_1), untoe(in(toe(q_1),q_2))\rangle = \langle untoe(q_1), in(toe(q_1), untoe(q_2))\rangle.$$

With the appropriate axioms for queues, the validity of these facts is easily verified.

For the $a=\tau$ we find that the precondition $a=\tau \Rightarrow g_a(d,e_1) \neq g_\tau(d,e_2)$ is instantiated to $\tau=\tau \Rightarrow \langle untoe(q_1), in(toe(q_1),q_2)\rangle \neq \langle untoe(q_1), in(toe(q_1),q_2)\rangle$, which is a trivial contradiction.

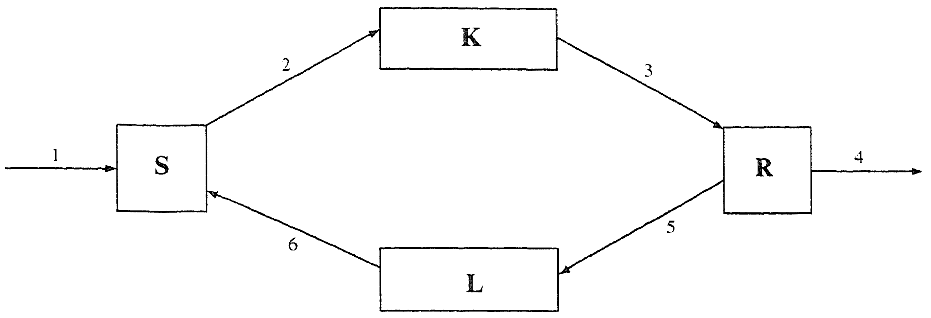


Fig. 6.

Now, by Theorem 7.4, the following TP-reduced version (see Definition 7.3) of $Q(\langle q_1,q_2\rangle)$ is branching bisimilar to $Q(\langle q_1,q_2\rangle)$.

$$\begin{aligned}
&Q_r(\langle q_1,q_2\rangle) = \\
&\textstyle\sum_{e_r:E_r} r(e_r) \cdot Q_r(\langle in(e_r,q_1),q_2\rangle) \lhd \top \wedge empty(q_1) \rhd \delta + \\
&\tau \cdot Q_r(\langle untoe(q_1), in(toe(q_1),q_2)\rangle) \lhd ne(q_1) \rhd \delta + \\
&s(toe(q_2)) \cdot Q_r(\langle q_1, untoe(q_2)\rangle) \lhd ne(q_2) \wedge empty(q_1) \rhd \delta
\end{aligned}$$

Note that after the TP-reduction q_1 never contains more than one element!

8.2. *The alternating bit protocol*

The alternating bit protocol (ABP) consists of a sender S, a receiver R and two unreliable channels K and L (see [3, p. 108]). All these components can straightforwardly be described by a linear process (see Fig. 6).

The sender

The variables d_s, b_s and n_s are the data parameter, the bit and the state of the sender. If $n_s = 0$ then S can read a fresh datum $r_1(x)$. If $n_s = 1$, it wants to send data to channel K and if $n_s = 2$ then S is waiting for an acknowledgement.

$$\begin{aligned}
S(d_s{:}\mathrm{D}, b_s{:}\mathbb{B}, n_s{:}\mathbb{N}) = &\textstyle\sum_{x:\mathrm{D}} r_1(x) \cdot S(x,b_s,1) \lhd n_s = 0 \rhd \delta + \\
&s_2(d_s,b_s) \cdot S(d_s,b_s,2) \lhd n_s = 1 \rhd \delta + \\
&r_6(\neg b_s) \cdot S(d_s,b_s,1) \lhd n_s = 2 \rhd \delta + \\
&r_6(ce) \cdot S(d_s,b_s,1) \lhd n_s = 2 \rhd \delta + \\
&r_6(b_s) \cdot S(d_s,\neg b_s,0) \lhd n_s = 2 \rhd \delta
\end{aligned}$$

Note that this linear process is not clustered because the last three summands of this linear process equation all perform the same action $r_6(..)$.

The channels

We provide linear process equations for the channels K and L. Again, the processes are not clustered. Analogously to the sender, d_k, b_k and n_k are the data parameter, the bit and the state of channel K. If $n_k = 0$ then K can read a datum. If $n_k = 1$ then K can choose to deliver the datum correctly ($n_k := 2$) or to lose the datum and report a checksum error ce ($n_k := 3$). After delivery of either message, K can read again.

$$\begin{aligned} K(d_k{:}\mathsf{D}, b_k{:}\mathbb{B}, n_k{:}\mathbb{N}) = \textstyle\sum_{x:\mathsf{D}} \sum_{y:\mathbb{B}} r_2(x,y) \cdot K(x,y,1) \lhd n_k = 0 \rhd \delta \; + \\ i \cdot K(d_k, b_k, 2) \lhd n_k = 1 \rhd \delta \; + \\ i \cdot K(d_k, b_k, 3) \lhd n_k = 1 \rhd \delta \; + \\ s_3(d_k, b_k) \cdot K(d_k, b_k, 0) \lhd n_k = 2 \rhd \delta \; + \\ s_3(ce) \cdot K(d_k, b_k, 0) \lhd n_k = 3 \rhd \delta \end{aligned}$$

The linear process equation for channel L is almost identical to the linear process equation for channel K we just gave. The only difference lies in the fact that channel L does not transport any data but only an acknowledging bit.

$$\begin{aligned} L(b_\ell{:}\mathbb{B}, n_\ell{:}\mathbb{N}) = \textstyle\sum_{y:\mathbb{B}} r_5(y) \cdot L(y,1) \lhd n_\ell = 0 \rhd \delta \; + \\ i \cdot L(b_\ell, 2) \lhd n_\ell = 1 \rhd \delta \; + \\ i \cdot L(b_\ell, 3) \lhd n_\ell = 1 \rhd \delta \; + \\ s_6(b_\ell) \cdot L(b_\ell, 0) \lhd n_\ell = 2 \rhd \delta \; + \\ s_6(ce) \cdot L(b_\ell, 0) \lhd n_\ell = 3 \rhd \delta \end{aligned}$$

The meaning of the parameters of L is exactly the same as the meaning of the corresponding parameters of K.

The receiver

The parameters d_r, b_r and n_r are the data, bit and state of the receiver, respectively. If $n_r = 0$ then R is waiting for data to arrive via channel K. If $n_r = 1$ then R wants to send an acknowledgement via channel L and if $n_r = 2$ then R is ready to execute action $s_4(d_r)$, i.e. deliver a datum.

$$\begin{aligned} R(d_r{:}\mathsf{D}, b_r{:}\mathbb{B}, n_r{:}\mathbb{N}) = \; & \textstyle\sum_{x:\mathsf{D}} r_3(x, b_r) \cdot R(x, b_r, 1) \lhd n_r = 0 \rhd \delta \; + \\ & r_3(ce) \cdot R(d_r, b_r, 1) \lhd n_r = 0 \rhd \delta \; + \\ & \textstyle\sum_{x:\mathsf{D}} r_3(x, \neg b_r) \cdot R(x, \neg b_r, 2) \lhd n_r = 0 \rhd \delta \; + \\ & s_5(b_r) \cdot R(d_r, b_r, 0) \lhd n_r = 1 \rhd \delta \; + \\ & s_4(d_r) \cdot R(d_r, b_r, 1) \lhd n_r = 2 \rhd \delta \end{aligned}$$

The parallel composition

The parallel composition $\partial_{\{r_i, s_i \mid i=2,3,5,6\}}(S \,\|\, K \,\|\, L \,\|\, R)$ can then be described by the following linear process, which is easily calculated from the four components S, K, L and R. In order to improve the readability we write $X[{}^a\!/_b]$ for the process that is obtained from $X(..)$ by replacing b by a. For example, in the linear process equation for the sender we could have written $S[{}^2\!/_{n_s}]$ instead of $S(d_s, b_s, 2)$ and so on.

$$\begin{array}{rcl}
\multicolumn{3}{l}{X(d_s,b_s,n_s,d_k,b_k,n_k,b_\ell,n_\ell,d_r,b_r,n_r) \;=} \\
\sum_{x:\mathsf{D}} r_1(x) \cdot X[{}^{x}/_{d_s}][{}^{1}/_{n_s}] & \lhd & n_s = 0 \;\rhd\; \delta \;+ \\
c_2(d_s,b_s) \cdot X[{}^{2}/_{n_s}][{}^{d_s}/_{d_k}][{}^{b_s}/_{b_k}][{}^{1}/_{n_k}] & \lhd & n_s = 1 \wedge n_k = 0 \;\rhd\; \delta \;+ \\
c_6(b_\ell) \cdot X[{}^{0}/_{n_s}][{}^{0}/_{n_\ell}][{}^{\neg b_s}/_{b_s}] & \lhd & b_\ell = b_s \wedge n_s = 2 \wedge n_\ell = 2 \;\rhd\; \delta \;+ \\
c_6(b_\ell) \cdot X[{}^{1}/_{n_s}][{}^{0}/_{n_\ell}] & \lhd & b_\ell \neq b_s \wedge n_s = 2 \wedge n_\ell = 2 \;\rhd\; \delta \;+ \\
c_6(ce) \cdot X[{}^{1}/_{n_s}][{}^{0}/_{n_\ell}] & \lhd & n_s = 2 \wedge n_\ell = 3 \;\rhd\; \delta \;+ \\
c_3(d_k,b_k) \cdot X[{}^{d_k}/_{d_r}][{}^{1}/_{n_r}][{}^{0}/_{n_k}] & \lhd & b_k = b_r \wedge n_r = 0 \wedge n_k = 2 \;\rhd\; \delta \;+ \\
c_3(d_k,b_k) \cdot X[{}^{d_k}/_{d_r}][{}^{2}/_{n_r}][{}^{0}/_{n_k}][{}^{\neg b_r}/_{b_r}] & \lhd & b_k \neq b_r \wedge n_r = 0 \wedge n_k = 2 \;\rhd\; \delta \;+ \\
c_3(ce) \cdot X[{}^{1}/_{n_r}][{}^{0}/_{n_k}] & \lhd & n_k = 3 \wedge n_r = 0 \;\rhd\; \delta \;+ \\
c_5(b_r) \cdot X[{}^{0}/_{n_r}][{}^{b_r}/_{b_\ell}][{}^{1}/_{n_\ell}] & \lhd & n_r = 1 \wedge n_\ell = 0 \;\rhd\; \delta \;+ \\
s_4(d_r) \cdot X[{}^{1}/_{n_r}] & \lhd & n_r = 2 \;\rhd\; \delta \;+ \\
i \cdot X[{}^{2}/_{n_k}] & \lhd & n_k = 1 \;\rhd\; \delta \;+ \\
i \cdot X[{}^{3}/_{n_k}] & \lhd & n_k = 1 \;\rhd\; \delta \;+ \\
i \cdot X[{}^{2}/_{n_\ell}] & \lhd & n_\ell = 1 \;\rhd\; \delta \;+ \\
i \cdot X[{}^{3}/_{n_\ell}] & \lhd & n_\ell = 1 \;\rhd\; \delta
\end{array}$$

We assume that initially the alternating bits of S and K are equal and unequal to the alternating bits of L and R. Furthermore we assume that initially all data parameters are equal and that all state parameters are 0. So all initial states are of the form $(d,b,0,d,b,0,\neg b,0,d,\neg b,0)$. In the sequel we abbreviate this state by $init(d,b)$.

Let $I = \{c_2, c_3, c_5, c_6, i\}$ then $\tau_I(X(init(d,b)))$ is not τ-well-founded. For instance,

$$\begin{tikzcd}
X(init(d,b)) \arrow[r, "r_1(x)"] & X(x,b,1,d,b,0,\neg b,0,d,\neg b,0) \arrow[d, "c_2(x,b)"'] \\
X(x,b,1,x,b,0,\neg b,0,d,\neg b,0) \arrow[r, "c_2(x,b)"] & X(x,b,2,x,b,1,\neg b,0,d,\neg b,0) \arrow[d, "i"'] \\
X(x,b,2,x,b,0,\neg b,3,d,\neg b,0) \arrow[u, "c_6(ce)"] & X(x,b,2,x,b,3,\neg b,0,d,\neg b,0) \arrow[d, "c_3(ce)"'] \\
X(x,b,2,x,b,0,\neg b,1,d,\neg b,0) \arrow[u, "i"] & X(x,b,2,x,b,0,\neg b,0,d,\neg b,1) \arrow[l, "c_5(\neg b)"']
\end{tikzcd}$$

will, after hiding, result in a τ-loop. This means that we cannot use Theorem 3.5 or Lemma 3.8 in order to derive property (1). Theorem 3.2 is also useless since $\mathcal{T}_I(X(init(d,b)))$ is obviously not strongly confluent. However, if we divide the τ-steps of $\mathcal{T}_I(X)$ in progressing and non-progressing ones, such that the result is $\tau_>$-well-founded, then we can apply Theorem 6.4. Let Y be the process obtained from $\mathcal{T}_I(X)$ by labelling those τ-steps that set n_k or n_ℓ to 3, with $<$ and all the other τ-steps with $>$.

$$
\begin{array}{rlclcl}
\multicolumn{6}{l}{Y(d_s,b_s,n_s,d_k,b_k,n_k,b_\ell,n_\ell,d_r,b_r,n_r) \; =} \\
\sum_{x:D} r_1(x) & \cdot\; Y[^x/_{d_s}][^1/_{n_s}] & \lhd & n_s = 0 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^2/_{n_s}][^{d_s}/_{d_k}][^{b_s}/_{b_k}][^1/_{n_k}] & \lhd & n_s = 1 \wedge n_k = 0 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^0/_{n_s}][^0/_{n_\ell}][^{\neg b_s}/_{b_s}] & \lhd & b_\ell = b_s \wedge n_s = 2 \wedge n_\ell = 2 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^1/_{n_s}][^0/_{n_\ell}] & \lhd & b_\ell \neq b_s \wedge n_s = 2 \wedge n_\ell = 2 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^1/_{n_s}][^0/_{n_\ell}] & \lhd & n_s = 2 \wedge n_\ell = 3 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^{d_k}/_{d_r}][^1/_{n_r}][^0/_{n_k}] & \lhd & b_k = b_r \wedge n_r = 0 \wedge n_k = 2 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^{d_k}/_{d_r}][^2/_{n_r}][^0/_{n_k}][^{\neg b_r}/_{b_r}] & \lhd & b_k \neq b_r \wedge n_r = 0 \wedge n_k = 2 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^1/_{n_r}][^0/_{n_k}] & \lhd & n_k = 3 \wedge n_r = 0 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^0/_{n_r}][^{b_r}/_{b_\ell}][^1/_{n_\ell}] & \lhd & n_r = 1 \wedge n_\ell = 0 & \rhd & \delta + \\
s_4(d_r) & \cdot\; Y[^1/_{n_r}] & \lhd & n_r = 2 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^2/_{n_k}] & \lhd & n_k = 1 & \rhd & \delta + \\
\tau_< & \cdot\; Y[^3/_{n_k}] & \lhd & n_k = 1 & \rhd & \delta + \\
\tau_> & \cdot\; Y[^2/_{n_\ell}] & \lhd & n_\ell = 1 & \rhd & \delta + \\
\tau_< & \cdot\; Y[^3/_{n_\ell}] & \lhd & n_\ell = 1 & \rhd & \delta
\end{array}
$$

In order to proceed we need the following lemma:

Lemma 8.1. *The following invariant properties hold in every state of* $Y(init(d,b))$.

(i) $n_k \neq 0 \rightarrow (n_s = 2 \wedge d_s = d_k \wedge b_s = b_k)$
(ii) $n_\ell \neq 0 \rightarrow (n_s = 2 \wedge n_r = 0 \wedge n_k = 0 \wedge b_r = b_\ell)$
(iii) $n_r \neq 0 \rightarrow (n_k = 0 \wedge n_s = 2)$
(iv) $n_r = 0 \rightarrow b_r = b_\ell$
(v) $b_s = b_r \rightarrow (d_s = d_r \wedge n_s \neq 0)$

Proof. Straightforward induction. □

Next we show that $Y(init(d,b))$ is weakly $>$-confluent. Since we formulated the conditions for weak $>$-confluence for *clustered* linear processes (Theorem 6.4) we

Table 1

	r_1	s_4	$\tau_<1$	$\tau_<2$	$\tau_>1$	$\tau_>2$	$\tau_>3$	$\tau_>4$	$\tau_>5$	$\tau_>6$	$\tau_>7$	$\tau_>8$	$\tau_>9$	$\tau_>10$
$\tau_>1$	×	○	×	○	■	×	×	×	×	×	×	○	×	○
$\tau_>2$	×	○	○	×	×	■	×	×	○	○	○	×	○	×
$\tau_>3$	×	○	○	×	×	×	■	×	○	○	○	×	○	×
$\tau_>4$	×	○	○	×	×	×	×	■	○	○	○	×	○	×
$\tau_>5$	○	×	×	○	×	○	○	○	■	×	×	×	×	○
$\tau_>6$	○	×	×	○	×	○	○	○	×	■	×	×	×	○
$\tau_>7$	○	×	×	○	×	○	○	○	×	×	■	×	×	○
$\tau_>8$	○	×	○	×	○	×	×	×	×	×	×	■	○	×
$\tau_>9$	○	○		○	×	○	○	○	×	×	×	○	■	○
$\tau_>10$	○	○	○		○	×	×	×	○	○	○	×	○	■

first formulate a clustered version Y' of Y.

$$
\begin{array}{l}
Y'(d_s,b_s,n_s,d_k,b_k,n_k,b_\ell,n_\ell,d_r,b_r,n_r) \; = \\
\sum_{x:\mathrm{D}} r_1(x)\cdot Y'\;[x/d_s][1/n_s] \quad \lhd \quad n_s=0 \quad \rhd \quad \delta \; + \\
\quad s_4(d_r)\cdot Y'\;[1/n_r] \quad \lhd \quad n_r=2 \quad \rhd \quad \delta \; + \\
\sum_{n:\mathbb{N}} \tau_< \cdot Y' \begin{cases} [3/n_k] & \text{if } n=1 \\ [3/n_\ell] & \text{if } n=2 \end{cases} \quad \lhd \quad \begin{bmatrix} (n_k=1\wedge n=1)\vee \\ (n_\ell=1\wedge n=2) \end{bmatrix} \quad \rhd \quad \delta \; + \\
\sum_{n:\mathbb{N}} \tau_> \cdot Y' \begin{cases}
[2/n_s][d_s/d_k][b_s/b_k][1/n_k] & \text{if } n=1 \\
[0/n_s][0/n_\ell][\neg b_s/b_s] & \text{if } n=2 \\
[1/n_s][0/n_\ell] & \text{if } n=3 \\
[1/n_s][0/n_\ell] & \text{if } n=4 \\
[d_k/d_r][1/n_r][0/n_k] & \text{if } n=5 \\
[d_k/d_r][2/n_r][0/n_k][\neg b_r/b_r] & \text{if } n=6 \\
[1/n_r][0/n_k] & \text{if } n=7 \\
[0/n_r][b_r/b_\ell][1/n_\ell] & \text{if } n=8 \\
[2/n_k] & \text{if } n=9 \\
[2/n_\ell] & \text{if } n=10
\end{cases} \quad \lhd \quad \begin{bmatrix}
(n_s=1\wedge n_k=0\wedge n=1)\vee \\
(b_\ell=b_s\wedge n_s=2\wedge n_\ell=2\wedge n=2)\vee \\
(b_\ell\neq b_s\wedge n_s=2\wedge n_\ell=2\wedge n=3)\vee \\
(n_s=2\wedge n_\ell=3\wedge n=4)\vee \\
(b_k=b_r\wedge n_r=0\wedge n_k=2\wedge n=5)\vee \\
(b_k\neq b_r\wedge n_r=0\wedge n_k=2\wedge n=6)\vee \\
(n_k=3\wedge n_r=0\wedge n=7)\vee \\
(n_r=1\wedge n_\ell=0\wedge n=8)\vee \\
(n_k=1\wedge n=9)\vee \\
(n_\ell=1\wedge n=10)
\end{bmatrix} \quad \rhd \quad \delta
\end{array}
$$

Lemma 8.2. *$Y'(init(d,b))$ is weakly $>$-confluent.*

Proof. Confluence must be checked for all $a\in Act$ and $e_1:E_a$, with respect to all $e_2:E_{\tau_>}$. We do this by a straightforward application of Theorem 6.4. We have distinguished 140 cases that have been listed in Table 1. For 68 cases, marked with a × in the table the condition $b_a(d,e_1)\wedge b_{\tau_>}(d,e_2)$ does not hold. In 10 cases, marked with a ■, the condition $a=\tau_> \Rightarrow g_a(d,e_1)\neq g_{\tau_>}(d,e_2)$ is violated. In 60 of the remaining 62 confluence is immediately clear from the fact that the substitutions do not affect each other (i.e. they are commutative). These cases are marked with a ○ in the table.

So, there are 2 cases left. Each case corresponds to the choice of a channel to corrupt the datum or not. We only treat the case $\tau_<1$ and $\tau_>9$.

We take σ_2 empty and $\sigma=\sigma_1$ using that $a=\tau_<$. So, e_3 is irrelevant. We distinguish the following two cases:

- Assume $b_r = b_s$. We take for $\sigma_1 = \langle \tau_>, 5\rangle$ and $\sigma_3 = \langle \tau_>, 7\rangle$. We must now check the three requirements

  $$\mathscr{B}_{g_a(d,e_1)}(\sigma_3), \mathscr{B}_{g_{\tau_>}(d,e_2)}(\sigma) \text{ and } \mathscr{G}_{g_a(d,e_1)}(\sigma_3) = \mathscr{G}_{g_{\tau_>}(d,e_2)}(\sigma)$$

  These boil down to the following three proof obligations, where the trivial ones have been omitted. All obligations follow from the invariant in Lemma 8.1 in a straightforward fashion as we know that $n_k = 1$.

  $$n_r = 0, b_k = b_r \wedge n_r = 0 \text{ and } d_k = d_r.$$

- In the case that $b_r \neq b_s$, we take σ_1 empty and $\sigma_3 = \langle \tau_>, 7\rangle\ \langle \tau_>, 8\rangle\ \langle \tau_>, 10\rangle\ \langle \tau_>, 3\rangle\ \langle \tau_>, 1\rangle\ \langle \tau_>, 9\rangle$. The three requirements now become

  $$n_r = 0 \wedge n_\ell = 0 \wedge b_r \neq b_s \wedge n_s = 2, \mathit{true} \text{ and}$$

  $$n_s = 2 \wedge b_r = b_\ell \wedge n_\ell = 0 \wedge n_r = 0 \wedge d_s = d_k \wedge b_s = b_k.$$

  These also follow from the invariant and the fact that $n_k = 1$. $\square$

The TP-reduction now prescribes that in all states where there is a progressing τ-step all other actions can be removed. In the cases where $n_k = 1$ or $n_\ell = 1$ this is applicable; we can remove the non-progressing τ-steps. By removing all transitions that have become unreachable in this way, we obtain the following simple process of which each state has only one outgoing transition. In particular, the channels have become reliable. One easily verifies that this process is convergent. For example, the natural number $3\cdot((-n_s) \bmod 3)+3\cdot((n_r-2) \bmod 3)-n_k-n_\ell+4$ decreases after each τ-step. By Theorem 7.4 this reduced process is branching (and hence also weakly) bisimilar to the alternating bit protocol.

$$
\begin{array}{llll}
Z(d_s,b_s,n_s,d_k,b_k,n_k,b_\ell,n_\ell,d_r,b_r,n_r) \ = & & & \\
\sum_{x:\mathsf{D}} r_1(x)\cdot Z\ [x/d_s][1/n_s] & \lhd & n_s = 0 & \rhd\ \delta\ + \\
\quad s_4(d_r)\cdot Z\ [1/n_r] & \lhd & n_r = 2 & \rhd\ \delta\ + \\
\sum_{n:\mathbb{N}} \tau_> \cdot Z \begin{cases} [2/n_s][d_s/d_k][b_s/b_k][1/n_k] & \text{if } n{=}1 \\ [0/n_s][0/n_\ell][\neg b_s/b_s] & \text{if } n{=}2 \\ [d_k/d_r][2/n_r][0/n_k][\neg b_r/b_r] & \text{if } n{=}6 \\ [0/n_r][b_r/b_\ell][1/n_\ell] & \text{if } n{=}8 \\ [2/n_k] & \text{if } n{=}9 \\ [2/n_\ell] & \text{if } n{=}10 \end{cases} & \lhd & \left[\begin{array}{c} (n_s{=}1 \wedge n_k{=}0 \wedge n{=}1)\vee \\ (b_\ell{=}b_s \wedge n_s{=}2 \wedge n_\ell{=}2 \wedge n{=}2)\vee \\ (b_k{\neq}b_r \wedge n_r{=}0 \wedge n_k{=}2 \wedge n{=}6)\vee \\ (n_r{=}1 \wedge n_\ell{=}0 \wedge n{=}8)\vee \\ (n_k{=}1 \wedge n{=}9)\vee \\ (n_\ell{=}1 \wedge n{=}10) \end{array}\right] & \rhd\ \delta
\end{array}
$$

Acknowledgements

We thank Marc Bezem for discussion in an early phase of the project and Frits Vaandrager for demonstrating the importance of confluence in the verification of a leader election protocol. Furthermore, we thank Jaco van de Pol and Wan Fokkink for comments on a draft version of this paper and Anne Dicky for her valuable suggestions on improvement of the presentation.

References

[1] D.J. Andrews, J.F. Groote and C.A. Middelburg, eds. *Proc. Internat. Workshop on Semantics of Specification Languages*, Utrecht, Netherlands, Workshops in Computing (Springer, Berlin, 1993).

[2] A. Arnold and A. Dicky, An algebraic characterization of transition system equivalences, *Inform. and Comput.* **82** (1989) 198–229.

[3] J.C.M. Baeten and J.W. Klop, eds., *Proc. 1st Conf. on Theories of Concurrency, CONCUR '90*, Amsterdam, The Netherlands, August 1990, Lecture Notes in Computer Science, Vol. 458 (Springer, Berlin, 1990).

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

[5] J.A. Bergstra and J.W. Klop, The algebra of recursively defined processes and the algebra of regular processes, in: *Proc. 11th ICALP*, Antwerp, Lecture Notes in Computer Science, Vol. 172 (Springer, Berlin, 1984) 82–95.

[6] M.A. Bezem and J.F. Groote, A correctness proof of a one-bit sliding window protocol in μCRL, *Comput. J.* **37** (1994) 289–307.

[7] M.A. Bezem and J.F. Groote, Invariants in process algebra with data, in: B. Jonsson and J. Parrow, eds., *Proc. 5th Conf. on Theories of Concurrency, CONCUR '94*, Uppsala, Sweden, August 1994, Lecture Notes in Computer Science, Vol. 836 (Springer, Berlin, 1994) 401–416.

[8] R. Gerth, R. Kuiper, D. Peled and W. Penczek. A partial order approach to branching time logic model checking. Computer Science Report 94/53, Department of Mathematics and Computer Science, Eindhoven University of Technology, December 1994.

[9] J.F. Groote and A. Ponse, Proof theory for μCRL: a language for processes with data, in Andrews et al. [1, pp. 231–250].

[10] J.F. Groote and A. Ponse, The syntax and semantics of μCRL, in: A. Ponse, C. Verhoef and S.F.M. van Vlijmen, eds., *Proc. 1st Workshop in the Algebra of Communicating Processes, ACP '94*, Utrecht, Netherlands, July 1994 (Springer, Berlin, 1994) 26–62.

[11] J.F. Groote and M.P.A. Sellink, Confluence for process verification, Tech. Report 137, Logic Group Preprint Series, Utrecht University, June 1995.

[12] J.F. Groote and J.C. van de Pol, A bounded retransmission protocol for large data packets. A case study in computer checked verification, Tech. Report 100, Logic Group Preprint Series, Utrecht University, October 1993.

[13] G.J. Holzmann and D. Peled, An improvement in formal verification, in: *Proc. FORTE 1994 Conf.*, Bern, Switzerland, 1994.

[14] J.W. Klop, Term rewriting systems, in: S. Abramsky, D.M. Gabbay and T.S.E. Maibaum, eds., *Handbook of Logic in Computer Science*, Vol. 2 (Oxford Science Publications, 1992) 1–116.

[15] H. Korver and J. Springintveld, A computer-checked verification of Milner's scheduler, in: M. Hagiya and J.C. Mitchel, eds., *Proc. 2nd Internat. Symp. on Theoretical Aspects of Computer Software, TACS '94*, Sendai, Japan, Lecture Notes in Computer Science, Vol. 789 (Springer, Berlin, 1994) 161–178.

[16] R. Milner, *A Calculus of Communicating Systems*, Lecture Notes in Computer Science, Vol. 92 (Springer, Berlin, 1980).

[17] H. Qin, Efficient verification of determinate processes, in: J.C.M. Baeten and J.F. Groote, eds., *Proc. 2nd Conf. on Theories of Concurrency, CONCUR '91*, Amsterdam, Netherlands, August 1991, Lecture Notes in Computer Science, Vol. 527 (Springer, Berlin, 1991) 471–494.

[18] M.P.A. Sellink, Verifying process algebra proofs in type theory, in Andrews et al. [1, pp. 315–339].

[19] F.W. Vaandrager, Uitwerking Tentamen Protocolverificatie, unpublished manuscript, 1994.

[20] R.J. van Glabbeek, The linear time – branching time spectrum, in Baeten and Klop [3, pp. 278–297].