Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 242 (2009) 3–17
www.elsevier.com/locate/entcs

On Convergence-sensitive Bisimulation and the Embedding of CCS in Timed CCS

Roberto M. Amadio[1,2]

Université Paris Diderot

Abstract

We propose a notion of convergence-sensitive bisimulation that is built just over the notions of (internal) reduction and of (static) context. In the framework of timed CCS, we characterise this notion of 'contextual' bisimulation via the usual labelled transition system. We also remark that it provides a suitable semantic framework for a fully abstract embedding of untimed processes into timed ones. Finally, we show that the notion can be refined to include sensitivity to divergence.

Keywords: Bisimulation, convergence, timed CCS.

1 Introduction

The main motivation for this work is to build a notion of convergence-sensitive bisimulation from first principles, namely from the notions of *internal reduction* and of (static) *context*. A secondary motivation is to understand how asynchronous/untimed behaviours can be embedded fully abstractly into synchronous/timed ones. Because the notion of convergence is very much connected to the notion of time, it seems that a convergence-sensitive bisimulation should find a natural application in a synchronous/timed context. Thus, in a nutshell, we are looking for an 'intuitive' semantic framework that spans both untimed/asynchronous and timed/synchronous models.

For the sake of simplicity we will place our discussion in the well-known framework of (timed) CCS. We assume the reader is familiar with CCS [10]. Timed CCS (TCCS) is a 'timed' version of CCS whose basic principle is that *time passes exactly when no internal computation is possible*. This notion of 'time' is inspired by

[1] PPS, UMR-7126. Work partially supported by ANR-06-SETI-010-02.
[2] Email: amadio@pps.jussieu.fr

doi:10.1016/j.entcs.2009.06.010

early work on the ESTEREL synchronous language [3], and it has been formalised in various dialects of CCS [14,12,6]. Here we shall follow the formalisation in [6].

As in CCS, one models the internal computation with an action τ while the passage of (discrete) time is represented by an action tick that implicitly synchronizes all the processes and moves the computation to the next instant. [3]

In this framework, the basic principle we mentioned is formalised as follows:

$$P \xrightarrow{\mathsf{tick}} \cdot \text{ iff } P \not\xrightarrow{\tau} \cdot$$

where we write $P \xrightarrow{\mu} \cdot$ if P can perform an action μ. TCCS is designed so that if P is a process built with the usual CCS operators and P cannot perform τ actions then $P \xrightarrow{\mathsf{tick}} P$. In other terms, CCS processes are *time insensitive*. To compensate for this property, one introduces a new binary operator $P \triangleright Q$, called *else_next*, that tries to run P in the current instant and, if it fails, runs Q in the following instant.

We assume countably many names $a, b, \ldots$ For each name a there is a communication action a and a co-action $\overline{a}$. We denote with $\alpha, \beta, \ldots$ the usual CCS actions which are composed of either an internal action τ or of a communication action $a, \overline{a}, \ldots$. We denote with $\mu, \mu', \ldots$ either an action α or the distinct action tick.

The TCCS processes $P, Q, \ldots$ are specified by the following grammar

$$P ::= 0 \mid a.P \mid P + P \mid P \mid P \mid \nu a\ P \mid A(\mathbf{a}) \mid P \triangleright P\ .$$

We denote with $fn(P)$ the names free in P. We adopt the usual convention that for each thread identifier A there is a unique defining equation $A(\mathbf{b}) = P$ where the parameters $\mathbf{b}$ include the names in $fn(P)$. The related labelled transition system is specified in table 1.

Say that a process is a CCS process if it does not contain the *else_next* operator. The reader can easily verify that:

(1) $P \xrightarrow{\mathsf{tick}} \cdot$ if and only if $P \not\xrightarrow{\tau} \cdot$.

(2) If $P \xrightarrow{\mathsf{tick}} Q_i$ for $i = 1, 2$ then $Q_1 = Q_2$. One says that the passage of time is *deterministic*.

(3) If P is a CCS process and $P \xrightarrow{\mathsf{tick}} Q$ then $P = Q$. Hence CCS processes are closed under labelled transitions.

It will be convenient to write $\tau.P$ for $\nu a\ (a.P \mid \overline{a}.0)$ where $a \notin fn(P)$, $\mathsf{tick}.P$ for $0 \triangleright P$, and Ω for the diverging process $\tau.\tau.\ldots$.

Remark 1.1 (1) In the labelled transition system in table 1, the definition of the tick action relies on the τ action and the latter relies on the communication actions $a, a', \ldots$. There is a well known method to give a direct definition of the τ action that does not refer to the communication actions. Namely, one defines (internal) reduction rules such as $(a.P + Q \mid \overline{a}.P' + Q') \to (P \mid P')$ which are applied modulo a suitable structural equivalence.

[3] There seems to be no standard terminology for this action. It is called ϵ in [14], χ in [12], σ in [6], and sometimes 'next' in 'synchronous' languages à la ESTEREL [2].

$$\frac{}{a.P \xrightarrow{a} P} \qquad \frac{P \xrightarrow{a} P' \quad Q \xrightarrow{\overline{a}} Q'}{(P \mid Q) \xrightarrow{\tau} (P' \mid Q')}$$

$$\frac{P \xrightarrow{\alpha} P'}{(P \mid Q) \xrightarrow{\alpha} (P' \mid Q)} \qquad \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}$$

$$\frac{A(\mathbf{a}) = P}{A(\mathbf{b}) \xrightarrow{\tau} [\mathbf{b}/\mathbf{a}]P} \qquad \frac{P \xrightarrow{\alpha} P'}{P \triangleright Q \xrightarrow{\alpha} P'}$$

$$0 \xrightarrow{\mathsf{tick}} 0 \qquad a.P \xrightarrow{\mathsf{tick}} a.P$$

$$\frac{P \not\xrightarrow{\tau} \cdot}{P \triangleright Q \xrightarrow{\mathsf{tick}} Q} \qquad \frac{(P_1 \mid P_2) \not\xrightarrow{\tau} \cdot \quad P_i \xrightarrow{\mathsf{tick}} Q_i \quad i = 1,2}{(P_1 \mid P_2) \xrightarrow{\mathsf{tick}} (Q_1 \mid Q_2)}$$

$$\frac{P_i \xrightarrow{\mathsf{tick}} Q_i \quad i = 1,2}{P_1 + P_2 \xrightarrow{\mathsf{tick}} Q_1 + Q_2} \qquad \frac{P \xrightarrow{\mu} Q \quad a, \overline{a} \neq \mu}{\nu a\ P \xrightarrow{\mu} \nu a\ Q}$$

Table 1
Labelled transition system

(2) The labelled transition system in table 1 relies on *negative* conditions of the shape $P \not\xrightarrow{\tau}$. These conditions can be replaced by a condition $\exists L\ P \downarrow L$, where L is a finite set of communication actions. The predicate '$\downarrow$' can be defined as follows:

$$0 \downarrow \emptyset \qquad a.P \downarrow \{a\} \qquad \frac{P_i \downarrow L_i, \quad i = 1,2}{(P_1 + P_2) \downarrow L_1 \cup L_2}$$

$$\frac{P \downarrow L}{P \triangleright Q \downarrow L} \qquad \frac{P \downarrow L}{(\nu a\ P) \downarrow L \backslash \{a, \overline{a}\}} \qquad \frac{P_i \downarrow L_i, \quad i = 1,2 \quad L_1 \cap \overline{L_2} = \emptyset}{(P_1 \mid P_2) \downarrow L_1 \cup L_2}$$

1.1 Signals and a deterministic fragment

As already mentioned, the TCCS model has been inspired by the notion of time available in the ESTEREL model [4] and its relatives such as SL [5]. These models rely on *signals* as the basic communication mechanism. Unlike a channel, a signal persists within the instant and disappears at the end of it. It turns out that a signal can be defined recursively in TCCS as:

$$\mathsf{emit}(a) = \overline{a}.\mathsf{emit}(a) \triangleright 0$$

The 'present' statement of SL that either reads a signal and continues the computation in the current instant or reacts to the absence of the signal in the following instant can be coded as follows:

$$\mathsf{present}\ a\ \mathsf{do}\ P\ \mathsf{else}\ Q = a.P \triangleright Q$$

Modulo these encodings, the resulting fragment of TCCS is specified as follows:

$$P ::= 0 \mid \mathsf{emit}(a) \parallel \mathsf{present}\ a\ \mathsf{do}\ P\ \mathsf{else}\ P \mid (P \mid P) \mid \nu a\ P \mid A(\mathbf{a})\ .$$

Notice that, unlike in (T)CCS, communication actions have an input or output polarity. The most important property of this fragment is that its processes are *deterministic* [5,1].

1.2 The usual labelled bisimulation

As usual, one can define a notion of *weak transition* as follows:

$$\stackrel{\mu}{\Rightarrow} = \begin{cases} (\stackrel{\tau}{\rightarrow})^* & \text{if } \mu = \tau \\ (\stackrel{\tau}{\rightarrow})^* \circ \stackrel{\mu}{\rightarrow} \circ (\stackrel{\tau}{\rightarrow})^* & \text{otherwise} \end{cases}$$

where the notation X^* stands for the reflexive and transitive closure of a binary relation X. When focusing just on internal reduction, we shall write $\rightarrow$ for $\stackrel{\tau}{\rightarrow}$ and $\Rightarrow$ for $\stackrel{\tau}{\Rightarrow}$. We write $P \rightarrow \cdot$ if $\exists\, P'\ (P \rightarrow P')$, otherwise we say that P has converged and write $P \downarrow$. We write $P \Downarrow$ if $\exists Q\ \ (P \Rightarrow Q$ and $Q \downarrow)$. Thus $P \Downarrow$ means that P *may* converge, *i.e.*, there is a reduction sequence to a process that has converged. Because $P \downarrow$ iff $P \stackrel{\mathsf{tick}}{\longrightarrow} \cdot$, we have that $P \Downarrow$ iff $P \stackrel{\mathsf{tick}}{\Rightarrow} \cdot$.

With respect to the notion of weak transition, one can define the usual notion of bisimulation as the largest symmetric relation $\mathcal{R}$ such that if $(P, Q) \in \mathcal{R}$ and $P \stackrel{\mu}{\Rightarrow} P'$ then for some Q', $Q \stackrel{\mu}{\Rightarrow} Q'$ and $(P', Q') \in \mathcal{R}$. We denote with $\approx^u$ the largest labelled bisimulation (u for *usual*). When looking at CCS processes, one may focus on CCS actions (thus excluding the tick action). We denote with $\approx^u_{ccs}$ the resulting labelled bisimulation.

1.3 CCS vs. TCCS

As we already noticed, TCCS has been designed so that CCS can be regarded as a transition closed subset of TCCS. A natural question is whether two CCS processes which are equivalent with respect to an untimed environment are still equivalent in a timed one. For instance, Milner [9] discusses a similar question when comparing CCS to SCCS. [4]

1.3.1 Testing semantics

In the context of TCCS and of a testing semantics, the question has been answered negatively by Hennessy and Regan [6]. For instance, they notice that the processes $P = a.(b + c.b) + a.(d + c.d)$ and $Q = a.(b + c.d) + a.(d + c.b)$ are 'untimed' testing equivalent but 'timed' testing inequivalent. The relevant test is the one that checks that if an action b cannot follow an action a in the current instant then an action b

[4] The notion of instant in SCCS is quite different from the one considered in TCCS/ESTEREL. In the former one declares explicitly what each thread does at each instant while in the latter the duration of an instant is the result of an arbitrarily complex interaction among the different threads.

will happen in the following instant just after an action c (process P will not pass this test while process Q does). This remark motivated the authors to develop a notion of 'timed' testing semantics.

1.3.2 Bisimulation semantics

What is the situation with the usual labelled bisimulation semantics recalled in section 1.2? Things are fine for *reactive* processes which are defined as follows.

Definition 1.2 A process P is reactive if whenever $P \stackrel{\mu_1}{\Rightarrow} \cdots \stackrel{\mu_n}{\Rightarrow} Q$, for $n \geq 0$, we have the property that all sequences of τ reductions starting from Q terminate.

Proposition 1.3 *Suppose P, Q are CCS reactive processes. Then $P \approx^u Q$ if and only if $P \approx^u_{ccs} Q$.*

Proof. Clearly, $\approx^u$ is a CCS bisimulation, hence $P \approx^u Q$ implies $P \approx^u_{ccs} Q$. To show the converse, we prove that $\approx^u_{ccs}$ is a timed bisimulation. So suppose $P \approx^u_{ccs} Q$ and $P \stackrel{\mathsf{tick}}{\Rightarrow} P'$. This means $P \stackrel{\tau}{\Rightarrow} P_1 \stackrel{\mathsf{tick}}{\longrightarrow} P_1 \stackrel{\tau}{\Rightarrow} P'$. Then for some Q_1, $Q \stackrel{\tau}{\Rightarrow} Q_1$ and $P_1 \approx^u_{ccs} Q_1$. Further, because Q_1 is reactive there is a Q_2 such that $Q_1 \stackrel{\tau}{\Rightarrow} Q_2$ and $Q_2 \downarrow$. By definition of bisimulation and the fact that $P_1 \downarrow$, we have that $P_1 \approx^u_{ccs} Q_2$. So for some Q', $Q_2 \stackrel{\tau}{\Rightarrow} Q'$ and $P' \approx^u_{ccs} Q'$. Thus we have shown that there is a Q' such that $Q \stackrel{\mathsf{tick}}{\Rightarrow} Q'$ and $P' \approx^u_{ccs} Q'$. □

Proposition 1.3 fails when we look at *non-reactive* processes. For instance, 0 and Ω are regarded as untimed equivalent but they are obviously timed inequivalent since the second process does not allow time to pass. This example suggests that if we want to extend proposition 1.3 to non-reactive processes, then the notion of bisimulation has to be *convergence sensitive*.

One possibility could be to adopt the usual bisimulation $\approx^u$ on CCS processes. We already noticed that if P is a CCS process and $P \stackrel{tick}{\longrightarrow} Q$ then $P = Q$. Thus in the bisimulation game between CCS processes, the condition '$P \stackrel{\mathsf{tick}}{\Rightarrow} P'$ implies $Q \stackrel{\mathsf{tick}}{\Rightarrow} Q'$' can be replaced by '$P \Downarrow$ implies $Q \Downarrow$'. The resulting equivalence on CCS processes is not new, for instance it appears in [8] as the so called *stable* weak bisimulation. One may notice that this equivalence has reasonably good congruence properties.

Proposition 1.4 *Suppose $P_1 \approx^u P_2$ and $Q_1 \approx^u Q_2$. Then*

(1) $(P_1 \mid R) \approx^u (P_2 \mid R)$.

(2) *If $P_1, P_2 \downarrow$ then $P_1 \triangleright Q_1 \approx^u P_2 \triangleright Q_2$.*

Proof. First note that we can work with an asymmetric definition of bisimulation where a strong transition is matched by a weak one.

(1) We just check the condition for the tick action. Suppose $(P_1 \mid R) \stackrel{\mathsf{tick}}{\longrightarrow} (P_1' \mid R')$. This entails $P_1 \stackrel{\mathsf{tick}}{\longrightarrow} P_1'$ and $R \stackrel{\mathsf{tick}}{\longrightarrow} R'$. Then $P_2 \stackrel{\tau}{\Rightarrow} P_2''$, $P_2'' \downarrow$, and $P_1 \approx^u P_1''$. Also $P_2'' \stackrel{\mathsf{tick}}{\Rightarrow} P_2'$ and $P_1' \approx^u P_2''$. Finally, we have that $(P_2'' \mid R) \downarrow$ because if they could synchronise on a name a then so could $(P_1 \mid R)$.

(2) There are two cases to consider. If $P_1 \triangleright Q_1 \xrightarrow{\mathsf{tick}} Q_1$ then $P_2 \triangleright Q_2 \xrightarrow{\mathsf{tick}} Q_2$. If $P_1 \triangleright Q_1 \xrightarrow{a} P_1'$ because $P_1 \xrightarrow{a} P_1'$ then $P_2 \stackrel{a}{\Rightarrow} P_2'$ and $P_1' \approx^u P_2'$. □

Remark 1.5 The *else_next* operator suffers from the same compositionality problems as the sum operator. For instance, $0 \approx^u \tau.0$ but $0 \triangleright Q = \mathsf{tick}.Q$ while $\tau.0 \triangleright Q \approx^u 0$. As for the sum operator, one may remark that in practice we are interested in a *guarded* form of the *else_next* operator. Namely, the *else_next* operator is only introduced as an alternative to a communication action (the *present* operator discussed in section 1.1 is such an example). Proposition 1.4(2) entails that in this form, the *else_next* operator preserves bisimulation equivalence.

1.3.3 An alternative path

The reader might have noticed that on CCS processes $\approx^u$ *refines* $\approx^u_{ccs}$ by adding may convergence as an observable along with the usual labelled transitions. This is actually the case of all convergence/divergence sensitive bisimulations we are aware of (see, *e.g.*, [15,8]). The question we wish to investigate is: what happens if we just take may convergence as an observable without assuming the observability of the labelled transitions? The question can be motivated by both pragmatic and mathematical considerations. On the pragmatic side, one may argue that the normal operation of a timed/synchronous program is to receive inputs at the beginning of each instant and to produce outputs at the end of each instant. Thus, unless the instant terminates, no observation is possible. For instance, the process $(a \mid \Omega)$ could be regarded as equivalent to Ω, while they are distinguished by the usual bisimulation $\approx^u$ on the ground that the labelled transition a is supposed to be directly observable.

On the mathematical side, it has been remarked by many authors that the notion of labelled transition system is not necessarily compelling. Specifically, one would like to define a notion of bisimulation without an *a priori* commitment to a notion of label. To cope with this problem, a well-known approach started in [11] and elaborated in [7] is to look at 'internal' reductions and at a basic notion of 'barb' and then to close under contexts thus producing a notion of 'contextual' bisimulation. However, even the notion 'barb' is not always easy to define and justify (an attempt based on the concept of *bi-orthogonality* is described in [13]). It seems to us that a natural approach which applies to a wide variety of formalisms is to regard convergence (may-termination) as the 'intrinsic' basic observable automatically provided by the internal reduction relation.

1.3.4 Contribution

Following these preliminary considerations, we are now in a position to describe our contribution.

(i) We introduce a notion of contextual bisimulation for CCS whose basic observable (or barb) is the may-termination predicate (section 2).

(ii) We provide various characterisations of this equivalence culminating in one based on a suitable 'convergence-sensitive' labelled bisimulation (section 3).

(iii) We derive from this characterisation that (section 4):
  (a) the embedding of CCS in TCCS is fully abstract (even for non-reactive processes).
  (b) the proposed equivalence coincides with the usual one on reactive processes.
  (c) on non-reactive processes it identifies more processes than the usual timed labelled bisimulation $\approx^u$.
  (d) on non-reactive CCS processes it is incomparable with the usual labelled CCS bisimulation $\approx^u_{ccs}$.

(iv) We refine the proposed notion of contextual bisimulation by making it sensitive to *divergence* and show that the characterisation results mentioned above can be extended to this case (section 5).

The development will take place in the context of so called *weak* bisimulation [10] which is more interesting and challenging than *strong* bisimulation.

2 Convergence sensitive bisimulation

We denote with $C, D, \ldots$ one hole *static contexts* specified by the following grammar:

$$C ::= [\,] \mid C \mid P \mid \nu a C$$

We require that the notion of bisimulation we consider is preserved by the static contexts in the sense of [7].

Definition 2.1 [bisimulation] A symmetric relation $\mathcal{R}$ on processes is a bisimulation if $P\mathcal{R}Q$ implies:

cxt for any static context C, $C[P]\mathcal{R}C[Q]$.

red $P \stackrel{\mu}{\Rightarrow} P'$, $\mu \in \{\tau, \mathsf{tick}\}$ implies $\exists Q'$ $(Q \stackrel{\mu}{\Rightarrow} Q'$ and $P'\mathcal{R}Q')$.

We denote with $\approx$ the largest bisimulation.

Remark 2.2 (1) In view of remark 1.1(1), the definition 2.1 of bisimulation does *not* assume the labels $a, a', \ldots$ which correspond to the communication action. Not only the labels are not considered in the bisimulation game, but they are not even required in the definition of the τ action. This means that the definition can be directly transferred to more complex process calculi where the definition of communication action is at best unclear.

(2) For CCS processes, if $P \xrightarrow{\mathsf{tick}} Q$ then $P = Q$. It follows that in the definition above, the condition **[red]** when $\mu = \mathsf{tick}$ can be replaced by $P \Downarrow$ implies $Q \Downarrow$. This is obviously false for processes including the *else_next* operator; in this case one needs the tick action to observe the behaviour of processes after the first instant, *e.g.*, to distinguish $\mathsf{tick}.a$ from $\mathsf{tick}.b$.

In view of the previous remark, the definition of bisimulation is specialised to CCS processes by simply restricting the condition **[cxt]** to CCS static contexts. We denote with $\approx_{ccs}$ the resulting largest bisimulation.

Next we remark that the observability of a 'stable commitment (or barb)' is entailed by the observation of convergence.

Definition 2.3 We say that P (stably) commits on a, and write $P \Downarrow_a$, if $P \Rightarrow P'$, $P' \downarrow$, and $P' \xrightarrow{a} \cdot$.[5]

Proposition 2.4 *If $P \approx Q$ and $P \Downarrow_a$ then $Q \Downarrow_a$.*

Proof. Suppose $P \Downarrow_a$ and $P \approx Q$. Then $P \Rightarrow P'$, $P' \downarrow$, and $P' \xrightarrow{a} \cdot$. By definition of bisimulation, $Q \Rightarrow Q''$ and $P' \approx Q''$. Moreover, $Q'' \Rightarrow Q'$, $Q' \downarrow$, $Q' \approx P' \approx Q''$. To show that $Q' \xrightarrow{a} \cdot$, consider the context $C = ([\,] \mid \overline{a}.\Omega)$. Then we have $C[P'] \not\Downarrow$, while $C[Q'] \Downarrow$ if and only if $Q' \not\xrightarrow{a} \cdot$. □

Another interesting notion is that of *contextual convergence*.

Definition 2.5 We say that a process P is contextual convergent, and write $P \Downarrow_C$, if $\exists\, C \;\; (C[P] \Downarrow)$.

Clearly, $P \Downarrow$ implies $P \Downarrow_C$ but the converse fails taking, for instance, $(a{+}b) \mid \overline{a}.\Omega$. Contextual convergence, can be characterised as follows.

Proposition 2.6 *The following conditions are equivalent:*

(1) $P \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_n} P'$ *and* $P' \downarrow$.

(2) *There is a CCS process Q such that* $(P \mid Q) \Downarrow$.

(3) $P \Downarrow_C$.

Proof. (1 ⇒ 2) Suppose $P_0 \xrightarrow{\alpha_1} P_1 \cdots \xrightarrow{\alpha_n} P_n$ and $P_n \downarrow$. We build the process Q in (2) by induction on n. If $n = 0$ we can take $Q = 0$. Otherwise, suppose $n > 0$. By inductive hypothesis, there is Q_1 such that $(P_1 \mid Q_1) \Downarrow$. We proceed by case analysis on the first action α_1. If $\alpha_1 = \tau$ take $Q = Q_1$ and if $\alpha_1 = a$ take $Q = \overline{a}.Q_1$.

(2 ⇒ 3) Taking the static context $C = [\,] \mid Q$.

(3 ⇒ 1) First, check by induction on a static context C that $P \xrightarrow{\tau} \cdot$ implies $C[P] \xrightarrow{\tau} \cdot$. Hence $C[P] \downarrow$ implies $P \downarrow$. Second, show that $C[P] \xrightarrow{\alpha} Q$ implies that $Q = C'[P']$ where C' is a static context and either $P = P'$ or $P \xrightarrow{\alpha'} P'$. Third, suppose $C[P] \xrightarrow{\tau} Q_1 \cdots \xrightarrow{\tau} Q_n$ with $Q_n \downarrow$. Show by induction on n that P can make a series of labelled transitions and reach a process which has converged. □

Remark 2.7 As shown by the characterisation above, the notion of contextual convergence is unchanged if we restrict our attention to contexts composed of CCS processes.

We notice that a bisimulation never identifies a process which is contextual convergent with one which is not while identifying all processes which are not contextual convergent.

[5] Note that in this definition the process 'commits' on action a only when it has converged.

Proposition 2.8 (1) *If $P \approx Q$ and $P \Downarrow_C$ then $Q \Downarrow_C$.*

(2) *If $P \not\Downarrow_C$ and $Q \not\Downarrow_C$ then $P \approx Q$.*

Proof. (1) If $P \Downarrow_C$ then for some context C, $C[P] \Downarrow$. By condition **[cxt]**, we have that $C[P] \approx C[Q]$, and by condition **[red]** we derive that $C[Q] \Downarrow$. Hence $Q \Downarrow_C$.

(2) We notice that the relation $S = \{(P, Q) \mid P, Q \not\Downarrow_C\}$ is a bisimulation. Indeed: (i) if $P \not\Downarrow_C$ then $C[P] \not\Downarrow_C$, (ii) if $P \Rightarrow P'$ and $P \not\Downarrow_C$ then $P' \not\Downarrow_C$, and (iii) if $P \not\Downarrow_C$ then $P \stackrel{\mathsf{tick}}{\not\Rightarrow} \cdot$. □

3 Characterisation

We characterise the (contextual and convergence sensitive) bisimulation introduced in definition 2.1 by means of a labelled bisimulation. The latter is obtained from the former by replacing condition **[cxt]** with a suitable condition **[lab]** on labelled transitions as defined in table 1.

Definition 3.1 [labelled bisimulation] A symmetric relation $\mathcal{R}$ on processes is a labelled bisimulation if $P\mathcal{R}Q$ implies:

lab if $P \Downarrow_C$ and $P \stackrel{a}{\Rightarrow} P'$ then $Q \stackrel{\alpha}{\Rightarrow} Q'$ and $P'\mathcal{R}Q'$ where $\alpha \in \{a, \tau\}$ and $\alpha = a$ if $P' \Downarrow_C$.

red if $P \stackrel{\mu}{\Rightarrow} P'$, $\mu \in \{\tau, \mathsf{tick}\}$ then $\exists\, Q'$ $(Q \stackrel{\mu}{\Rightarrow} Q'$ and $P'\mathcal{R}Q')$.

We denote with $\approx^{\ell}$ the largest labelled bisimulation.

Remark 3.2 (1) By remark 2.2, on CCS processes the condition **[red]** when $\mu = \mathsf{tick}$ is equivalent to: '$P \Downarrow$ implies $Q \Downarrow$'. By remark 2.7, the notion of contextual convergence is unaffected if we restrict our attention to CCS processes. This means that, by definition, the (timed) labelled bisimulation restricted to CCS processes is the same as the labelled bisimulation on (untimed) CCS processes.

(2) The predicate of contextual convergence $\Downarrow_C$ plays an important role in the condition **[lab]**. To see why, suppose we replace it with the predicate $\Downarrow$ and assume we denote with $\approx^{\ell\Downarrow}$ the resulting largest labelled bisimulation. The following example shows that $\approx^{\ell\Downarrow}$ is not preserved by parallel composition. Consider:

$$P_1 = a.(b + c), \quad P_2 = a.b + a.c, \quad Q = \overline{a}.(d + \Omega)\ .$$

Then $(P_1 \mid Q) \approx^{\ell\Downarrow} (P_2 \mid Q)$ because both processes fail to converge. On the other hand, $(P_1 \mid Q) \mid \overline{d} \not\approx^{\ell\Downarrow} (P_2 \mid Q) \mid \overline{d}$ because the first may converge to $(b + c)$ which cannot be matched by the second process.

(3) One may consider an asymmetric and equivalent definition of labelled bisimulation where strong transitions are matched by weak transitions. To check the equivalence, it is useful to note that $P \not\Downarrow_C$ and $P \stackrel{\alpha}{\rightarrow} P'$ implies $P' \not\Downarrow_C$.

We provide a rather standard proof that bisimulation and labelled bisimulation coincide.

Proposition 3.3 *If $P \approx Q$ then $P \approx^{\ell} Q$.*

Proof. We show that the bisimulation $\approx$ is a labelled bisimulation. We denote with $P \oplus Q$ the internal choice between P and Q which is definable, *e.g.*, as $\tau.P + \tau.Q$. Suppose $P \Downarrow_C$ and $P \stackrel{a}{\Rightarrow} P'$. We consider a context $C = [\,] \mid T$ where $T = \overline{a}.((b \oplus 0) \oplus c)$ and b, c are 'fresh names' (not occurring in P, Q). We know $C[P] \approx C[Q]$ and $C[P] \Rightarrow (P' \mid (b \oplus 0))$. Thus $C[Q] \Rightarrow (Q' \mid T')$ where either $Q \stackrel{a}{\Rightarrow} Q'$ and $T \stackrel{\overline{a}}{\Rightarrow} T'$ or $Q \Rightarrow Q'$ and $T = T'$.

- Suppose $P' \not\Downarrow_C$. Then $(P' \mid (b \oplus 0)) \not\Downarrow_C$ and, by proposition 2.8, $(Q' \mid T') \not\Downarrow_C$. The latter implies that $Q' \not\Downarrow_C$. By contradiction, suppose $Q' \Downarrow_C$, that is $(Q' \mid R) \Downarrow$. Then $(Q' \mid T') \mid R \mid \overline{T'} \Downarrow$ (contradiction!), where we take $\overline{T'} = \overline{a}$ if $T' = T$ and $\overline{T'} = 0$ otherwise. Hence, $P' \approx Q'$ as required.

- Suppose $P' \Downarrow_C$. If $Q \stackrel{a}{\Rightarrow} Q'$ and $T \stackrel{\overline{a}}{\Rightarrow} T'$ then we show that it must be that $T' = (b \oplus 0)$. This is because if $P' \Downarrow_C$ then $P' \mid (b \oplus 0) \Downarrow_C$ which in turn implies that for some R (not containing the names b or c), $(P' \mid (b \oplus 0) \mid R) \Downarrow_b$. By proposition 2.4, we must have $Q'' = (Q' \mid T') \mid R \Downarrow_b$. Thus T' cannot be 0 and it cannot be $(b \oplus 0) \oplus c$, for otherwise $Q'' \Downarrow_c$ which cannot be matched by $(P' \mid (b \oplus 0) \mid R)$. Further, we have $P' \mid (b \oplus 0) \stackrel{\tau}{\rightarrow} P' \mid 0\ (= P')$. So $(Q' \mid (b \oplus 0)) \stackrel{\tau}{\Rightarrow} (Q' \mid T'')$ and $P' \approx (Q' \mid T'')$. The latter entails that $T'' = 0$.

  On the other hand, we show that $Q \stackrel{\tau}{\Rightarrow} Q'$ and $T = T'$ is impossible. Reasoning as above, we have $(P' \mid (b \oplus 0) \mid R) \Downarrow_b$. But then if $(Q' \mid T) \mid R \Downarrow_b$ we shall also have $(Q' \mid T) \mid R \Downarrow_c$. □

The following lemma relates contextual convergence to labelled bisimulation (cf. the similar proposition 2.8).

Lemma 3.4 (1) *If $P \approx^{\ell} Q$ and $P \Downarrow_C$ then $Q \Downarrow_C$.*

(2) *If $P \not\Downarrow_C$ and $Q \not\Downarrow_C$ then $P \approx^{\ell} Q$.*

Proof. (1) By proposition 2.6, if $P \Downarrow_C$ then $P \stackrel{\alpha_1}{\longrightarrow} \cdots \stackrel{\alpha_n}{\longrightarrow} P'$ and $P' \downarrow$. By definition of labelled bisimulation we should have $Q \stackrel{\alpha_1}{\Rightarrow} \cdots \stackrel{\alpha_n}{\Rightarrow} Q'$ and $P' \approx^{\ell} Q'$. Then $P' \stackrel{\mathsf{tick}}{\Rightarrow} \cdot$ entails $Q' \stackrel{\mathsf{tick}}{\Rightarrow}$, and therefore $Q \Downarrow_C$.

(2) By proposition 2.8, $P, Q \not\Downarrow_C$ implies $P \approx Q$, and by proposition 3.3 we conclude that $P \approx^{\ell} Q$. □

Proposition 3.5 *If $P \approx^{\ell} Q$ then $P \approx Q$.*

Proof. We show that labelled bisimulation is preserved by static contexts. In view of remark 3.2(3), we shall work with an asymmetric definition of bisimulation. With respect to this definition, we show that the following relations are labelled bisimulations:

$$\{(\nu a\ P, \nu a\ Q) \mid P \approx^{\ell} Q\} \cup \approx^{\ell}, \qquad \{(P \mid R, Q \mid R) \mid P \approx^{\ell} Q\} \cup \approx^{\ell}.$$

The case for restriction is a routine verification so we focus on parallel composition.

Suppose $(P \mid R) \xrightarrow{\mu} \cdot$. We proceed by case analysis.

- $(P \mid R) \xrightarrow{\alpha} (P \mid R')$ because $R \xrightarrow{\alpha} R'$. Then $(Q \mid R) \xrightarrow{\alpha} (Q \mid R')$.

- $(P \mid R) \xrightarrow{\mathsf{tick}} (P' \mid R')$ because $P \xrightarrow{\mathsf{tick}} P'$ and $R \xrightarrow{\mathsf{tick}} R'$. Then $Q \Rightarrow Q_1 \xrightarrow{\mathsf{tick}} Q_2 \Rightarrow Q'$ and $P' \approx^{\ell} Q'$. Notice that $P \approx^{\ell} Q_1$ with $P, Q_1 \downarrow$, and therefore $(Q_1 \mid R) \xrightarrow{\mathsf{tick}} (Q_2 \mid R')$. Hence $(Q \mid R) \stackrel{\mathsf{tick}}{\Rightarrow} (Q' \mid R')$.

- Suppose $(P \mid R) \Downarrow_C$ and $(P \mid R) \xrightarrow{a} (P' \mid R)$ because $P \xrightarrow{a} P'$. Then $P \Downarrow_C$ and therefore $Q \stackrel{\alpha}{\Rightarrow} Q'$, $\alpha \in \{a, \tau\}$, and $P' \approx^{\ell} Q'$. If $(P' \mid R) \Downarrow_C$ then $P' \Downarrow_C$ and this entails $\alpha = a$.

- Suppose $(P \mid R) \xrightarrow{\tau} (P' \mid R)$ because $P \xrightarrow{\tau} P'$. Then $Q \stackrel{\tau}{\Rightarrow} Q'$ and $P' \approx^{\ell} Q'$.

- Suppose $(P \mid R) \xrightarrow{\tau} (P' \mid R')$ because $P \xrightarrow{a} P'$ and $R \xrightarrow{\overline{a}} R'$. If $P, P' \Downarrow_C$ then $Q \stackrel{a}{\Rightarrow} Q'$ and $P' \approx^{\ell} Q'$. If $P \Downarrow_C$ and $P' \not\Downarrow_C$ then $Q \stackrel{\alpha}{\Rightarrow} Q'$, $\alpha \in \{a, \tau\}$, and $P' \approx^{\ell} Q'$. But then $(P' \mid R), (Q' \mid R) \not\Downarrow_C$, and we apply lemma 3.4. If $P \not\Downarrow_C$ then $Q \not\Downarrow_C$ and therefore $(Q \mid R) \not\Downarrow_C$, and we apply again lemma 3.4. □

As a first application of the characterisation we check that bisimulation is preserved by the *else_next* operator in the sense of proposition 1.4(2).

Corollary 3.6 *Suppose $P_1 \approx P_2$, $P_1, P_2 \downarrow$, and $Q_1 \approx Q_2$. Then $P_1 \triangleright Q_1 \approx P_2 \triangleright Q_2$.*

Proof. There are two cases to consider. If $P_1 \triangleright Q_1 \xrightarrow{\mathsf{tick}} Q_1$ then $P_2 \triangleright Q_2 \xrightarrow{\mathsf{tick}} Q_2$. If $P_1 \triangleright Q_1 \xrightarrow{a} P_1'$ because $P_1 \xrightarrow{a} P_1'$ then $P_2 \stackrel{a}{\Rightarrow} P_2'$, $P_1' \approx^{\ell} P_2'$, and $\alpha \in \{\tau, a\}$. We note that it must be that $\alpha = a$. Indeed, if $\alpha = \tau$ then since $P_2 \downarrow$ we must have $P_2' = P_2$ and $P_1' \Downarrow_C$. The latter forces $\alpha = a$ which is a contradiction. □

4 Embedding CCS in TCCS

In this section we collect some easy corollaries of the characterisation. First, we remark that two CCS processes are bisimilar when observed in an untimed/asynchronous environment if and only if they are bisimilar in a timed/synchronous environment.

Proposition 4.1 *Suppose P, Q are CCS processes. Then $P \approx Q$ if and only if $P \approx_{ccs} Q$.*

Proof. By propositions 3.3 and 3.5 we know that $\approx = \approx^{\ell}$. By remark 3.2(1), the labelled bisimulation on untimed processes coincides with the restriction to CCS processes of the timed labelled bisimulation. □

Second, we compare the notion of convergence-sensitive bisimulation we have introduced with the usual one we have recalled in the section 1.2. All the notions collapse on reactive processes.

Proposition 4.2 *Suppose P, Q are reactive processes. Then $P \approx Q$ if and only if $P \approx^{u} Q$.*

Proof. We know that $\approx = \approx^{\ell}$. Reactive processes are closed under labelled transitions and on reactive processes the conditions that define labelled bisimulation coincide with the ones for the usual bisimulation. □

The situation on non-reactive processes is summarised as follows where all implications are strict.

Proposition 4.3 *Suppose P, Q are processes.*

(1) *If $P \approx^{u} Q$ then $P \approx Q$.*

(2) *If moreover P and Q are CCS processes then $P \approx^{u} Q$ implies both $P \approx^{u}_{ccs} Q$ and $P \approx Q$.*

Proof. (1) The clauses in the definition of $\approx^{u}$ imply directly those in the definition of the labelled bisimulation that characterises $\approx$ (definition 3.1). To see that the converse fails note that $(a \mid \Omega) \approx \Omega$ while $(a \mid \Omega) \not\approx^{u} \Omega$.

(2) Use (1) and the fact that the clauses in the definition of $\approx^{u}$ imply directly those in the definition of $\approx^{u}_{ccs}$. To see that the converse fails use the counter-example in (1) and the fact that $0 \approx^{u}_{ccs} \Omega$ while $0 \not\approx^{u} \Omega$. □

5 Divergence sensitive bisimulation

We refine the notion of bisimulation to make it sensitive to *divergence* and show that the characterisation presented in section 3 can be adapted to this case.

We say that a process P may diverge and write $P \Uparrow$ if there is an infinite reduction sequence of τ actions that starts from P. We refine the notion of bisimulation by making it sensitive to divergence.

Definition 5.1 [$\Uparrow$-bisimulation] A symmetric relation $\mathcal{R}$ on processes is a divergence sensitive bisimulation ($\Uparrow$-bisimulation, for short) if it is a bisimulation according to definition 2.1 and if $P \mathcal{R} Q$ and $P \Uparrow$ implies $Q \Uparrow$. We denote with $\approx_{\Uparrow}$ the largest $\Uparrow$-bisimulation.

Remark 5.2 Say that a process P is strongly normalising if all reduction sequences of τ-actions that start from P terminate. A process is strongly normalising if and only if it may not diverge. It follows that one can give an equivalent formulation of $\Uparrow$-bisimulation by replacing the may divergence predicate with the strong normalisation predicate.

We notice the following properties whose proof is direct.

Proposition 5.3 (1) *If $P \approx_{\Uparrow} Q$ then $P \approx Q$.*

(2) *If $P \approx_{\Uparrow} Q$ and $P \Downarrow_{a}$ then $Q \Downarrow_{a}$.*

(3) *If $P \approx_{\Uparrow} Q$ and $P \Downarrow_{C}$ then $Q \Downarrow_{C}$.*

(4) *If $P \not\Downarrow_{C}$ then $P \Uparrow$.*

(5) *If $P \not\Downarrow_{C}$ and $Q \not\Downarrow_{C}$ then $P \approx_{\Uparrow} Q$.*

PROOF. (1) A $\Uparrow$-bisimulation is also a bisimulation.

(2) We apply (1) and proposition 2.4.

(3) We apply (1) and proposition 2.8(1).

(4) Immediate, by definition.

(5) If $P \not\Downarrow_C$ and $Q \not\Downarrow_C$ then $P \Uparrow$ and $Q \Uparrow$. □

It follows that $\Uparrow$-bisimulation coincides with bisimulation on the processes that are not contextual convergent. On the other hand, on those that are contextual convergent, it is a strictly finer notion as, *e.g.*, it distinguishes 0 from $A = \tau.A + \tau.0$.

The characterisation of $\Uparrow$-bisimulation turns out to be straightforward: it is enough to make the labelled bisimulation we have introduced in definition 3.1 sensitive to divergence.

Definition 5.4 [$\Uparrow$-labelled bisimulation] A symmetric relation $\mathcal{R}$ on processes is a divergence sensitive labelled bisimulation (or $\Uparrow$-labelled bisimulation) if it is a labelled bisimulation and if $P\mathcal{R}Q$ and $P \Uparrow$ implies that $Q \Uparrow$. We denote with $\approx^{\ell}_{\Uparrow}$ the largest $\Uparrow$-labelled bisimulation.

Because of the properties stated in proposition 5.3, one can repeat the proofs in section 3 while adding specific arguments to take the sensitivity to divergence into account.

Proposition 5.5 *If $P \approx_{\Uparrow} Q$ then $P \approx^{\ell}_{\Uparrow} Q$.*

PROOF. We show that $\approx_{\Uparrow}$ is a $\Uparrow$-labelled bisimulation by repeating the proof schema in proposition 3.3. Note that the condition that refers to divergence is the same for $\Uparrow$-bisimulation and for $\Uparrow$-labelled bisimulation. □

Lemma 5.6 (1) *If $P \approx^{\ell}_{\Uparrow} Q$ and $P \Downarrow_C$ then $Q \Downarrow_C$.*

(2) *If $P \not\Downarrow_C$ and $Q \not\Downarrow_C$ then $P \approx^{\ell}_{\Uparrow} Q$.*

PROOF. (1) Note that $P \approx^{\ell}_{\Uparrow} Q$ implies $P \approx^{\ell} Q$ and apply lemma 3.4(1).

(2) By proposition 5.3(5), $P \not\Downarrow_C$ and $Q \not\Downarrow_C$ implies $P \approx_{\Uparrow} Q$ and by proposition 5.5 the latter implies $P \approx^{\ell}_{\Uparrow} Q$. □

Proposition 5.7 *If $P \approx^{\ell}_{\Uparrow} Q$ then $P \approx_{\Uparrow} Q$.*

PROOF. As in proposition 3.5, we have to verify that $\approx^{\ell}_{\Uparrow}$ is preserved by name generation and parallel composition. For the former we note that $\nu a\ P \Uparrow$ if and only if $P \Uparrow$. For the latter, we can repeat the proof in proposition 3.5. Moreover, we have to consider the case where $P \approx^{\ell}_{\Uparrow} Q$ and $(P \mid R) \Uparrow$. The process $(P \mid R)$ diverges because: either P and R may engage in a finite number of synchronisations after which one of the two diverges or P and R may engage in an infinite number of synchronisations. Suppose the finite or infinite number of synchronisations between

P and R correspond to the transitions $P \stackrel{a_1}{\Rightarrow} P_1 \stackrel{a_2}{\Rightarrow} \cdots$ and $R \stackrel{\overline{a}_1}{\Rightarrow} R_1 \stackrel{\overline{a}_2}{\Rightarrow} \cdots$ If $P, P_1, \cdots$ are all contextually convergent then $Q \stackrel{a_1}{\Rightarrow} Q_1 \stackrel{a_2}{\Rightarrow} \cdots$ and $P_i \approx^{\ell}_{\Uparrow} Q_i$. Hence $(Q \mid R) \Uparrow$. If $P \Downarrow_C$ then $Q \Downarrow_C$ implies $(Q \mid R) \Downarrow_C$ which implies $(Q \mid R) \Uparrow$. Finally, suppose P_i is the least i such that $P_i \not\Downarrow_C$. Then $Q \stackrel{a_1}{\Rightarrow} \cdots \stackrel{a_{i-1}}{\Rightarrow} Q_{i-1} \stackrel{\alpha_i}{\Rightarrow} Q_i$ with $Q_i \not\Downarrow_C$ and $\alpha_i \in \{a_i, \tau\}$. If $\alpha_i = a_i$ then $(Q \mid R) \Uparrow$ because $(Q \mid R) \stackrel{\tau}{\Rightarrow} (Q_i \mid R')$ and $Q_i \Uparrow$. If $\alpha_i = \tau$ then $(Q \mid R) \Uparrow$ because $(Q \mid R) \stackrel{\tau}{\Rightarrow} (Q_i \mid R_{i-1})$ and $Q_i \Uparrow$. □

6 Conclusion

We have presented a natural notion of contextual and convergence sensitive bisimulation and we have shown that it can be characterised by a variant of the usual notion of labelled bisimulation relying on the concept of contextual convergence. As a direct corollary of this characterisation, we have shown that (untimed) CCS processes are embedded fully abstractly into timed ones. Finally, we have refined the notion of bisimulation to make it divergence-sensitive.

We believe that our main contribution, if any, is of a methodological nature. The notion of bisimulation we have introduced just requires the notions of reduction and static context as opposed to previous approaches that build on the notion of 'labelled' transition or on the notion of 'barb'. It would be interesting to apply the proposed approach to other situations where the notion of equivalence is unclear. For instance, we expect that our results can be extended to a TCCS with 'asynchronous' communication or with 'signal-based' communication.

References

[1] R. Amadio. The SL synchronous language, revisited. *Journal of Logic and Algebraic Programming*, 70:121-150, 2007.

[2] R. Amadio. A synchronous π-calculus. *Information and Computation*, 205(9):1470–1490, 2007.

[3] G. Berry, L. Cosserat. The Esterel synchronous programming language and its mathematical semantics. INRIA technical report 842, Sophia-Antipolis, 1988.

[4] G. Berry and G. Gonthier. The Esterel synchronous programming language. *Science of computer programming*, 19(2):87–152, 1992.

[5] F. Boussinot and R. De Simone. The SL synchronous language. *IEEE Trans. on Software Engineering*, 22(4):256–266, 1996.

[6] M. Hennessy, T. Regan. A process algebra of timed systems. *Information and Computation*, 117(2):221-239, 1995.

[7] K. Honda and N. Yoshida. On reduction-based process semantics. *Theoretical Computer Science*, 151(2):437-486, 1995.

[8] M. Lohrey, P. D'Argenio, and H. Hermanns: Axiomatising divergence. In Proc. ICALP, SLNCS 2380:585-596, 2002.

[9] R. Milner. Calculi for synchrony and asynchrony. *Theoretical Computer Science*, 25(3):267–310, 1983.

[10] R. Milner. Communication and Concurrency. Prentice-Hall, 1989.

[11] R. Milner and D. Sangiorgi. Barbed bisimulation. In Proc. *ICALP*, SLNCS 623:685–695, 1992.

[12] X. Nicolin, J. Sifakis. The algebra of timed processes (ATP): theory and application. *Information and Computation*, 114(1):131-178, 1994.

[13] J. Rathke, V. Sassone and P. Sobocinski. Semantic barbs and biorthogonality. In Proc. FoSSaCS 2007, SLNCS 4423:302-316, 2007.

[14] W. Yi. A calculus of real time systems. PhD thesis. Chalmers University, 1991.

[15] D. Walker. Bisimulation and divergence. Information and Computation, 85:202-241, 1990.