# A Faster–than Relation for Asynchronous Processes*

Gerald Lüttgen[1] and Walter Vogler[2]

[1] Department of Computer Science, Sheffield University, 211 Portobello Street, Sheffield S1 4DP, U.K., g.luettgen@dcs.shef.ac.uk
[2] Institut für Informatik, Universität Augsburg, D–86135 Augsburg, Germany, vogler@informatik.uni-augsburg.de

**Abstract.** This paper introduces a novel (bi)simulation–based faster–than preorder which relates asynchronous processes with respect to their worst–case timing behavior. The studies are conducted for a conservative extension of the process algebra CCS, called TACS, which permits the specification of maximal time bounds of actions. The most unusual contribution is in showing that the proposed faster–than preorder coincides with two other preorders, one of which considers the absolute times at which actions occur in system runs. The paper also develops the semantic theory of TACS, addressing congruence properties, equational laws, and abstractions from internal actions.

## 1 Introduction

*Process algebras* [5] provide a widely studied framework for reasoning about the behavior of concurrent systems. Early approaches, including Milner's CCS [15], focused on semantic issues of asynchronous processes, where the relative speeds between processes running in parallel is not bounded, i.e., one process may be arbitrarily slower or faster than another. This leads to a simple and mathematically elegant semantic theory analyzing the functional behavior of systems regarding their causal interactions with their environments. To include time as an aspect of system behavior, *timed process algebras* [4] were introduced. They usually model synchronous systems where processes running in parallel are under the regime of a common global clock and have a fixed speed. A well–known representative of discrete timed process algebras is Hennessy and Regan's TPL [11] which extends CCS by a timeout operator and a clock prefix demanding that exactly one time unit *must* pass before activating the argument process. Research papers on timed process algebras usually do not relate processes with respect to speed; the most notable exception is work by Moller and Tofts [17] which considers a faster–than preorder within a CCS–based setting, where processes are attached with lower time bounds. In practice, however, often upper time bounds are known to a system designer, determining how long a process may delay its execution. These can be used to compare the *worst–case timing behavior* of processes. The assumption

* Research support was partly provided under NASA Contract No. NAS1–97046.

of upper time bounds for *asynchronous* processes is exploited in distributed algorithms and was studied by the second author [6, 13, 12, 20, 21, 22] in settings equipped with DeNicola and Hennessy's testing semantics [9]. We re–emphasize that, in our context, "asynchronous" means that the relative speeds of system components are indeterminate.

In this paper we develop a novel (bi)simulation–based approach to compare asynchronous systems with respect to their worst–case timing behavior. To do so, we extend CCS by a rather specific notion of clock prefixing "$\sigma.$", where $\sigma$ stands for one time unit or a single clock tick. In contrast to TPL we interpret $\sigma.P$ as a process which *may* delay *at most* one time unit before executing $P$. Similar to TPL we view the occurrence of actions as instantaneous. This results in a new process algebra extending CCS, to which we refer as *Timed Asynchronous Communicating Systems* (TACS). To make our intuition of upper bound delays precise, consider the processes $\sigma.a.\mathbf{0}$ and $a.\mathbf{0}$, where $a$ denotes an action as in CCS. While the former process may delay an *enabled* communication on $a$ by one time unit, the latter must engage in the communication, i.e., $a$ is *non–urgent* in $\sigma.a.\mathbf{0}$ but *urgent* in $a.\mathbf{0}$. However, if a communication on $a$ is not enabled, then process $a.\mathbf{0}$ may wait until some communication partner is ready. To enforce a communication resulting in the internal action $\tau$, a time step in TACS is preempted by an urgent $\tau$. This is similar to timed process algebras employing the *maximal progress assumption* [11] where, however, in contrast to TACS, any internal computation is considered to be urgent. For TACS we introduce a *faster–than preorder* which exploits upper time bounds: a process is faster than another if both are linked by a relation which is a strong bisimulation for actions and a simulation for time steps.

The main contribution of this paper is the formal underpinning of our preorder, justifying why it is a good candidate for a faster–than relation on processes. There are at least two very appealing alternative definitions for such a preorder. First, one could allow the slower process to perform extra time steps when simulating an action or time step of the faster process. Second is the question of how exactly the faster process can match a time step and the subsequent behavior of the slower one. For illustrating this issue, consider the runs $a\sigma\sigma b$ and $\sigma a\sigma b$ which might be exhibited by some processes. One can argue that the first run is faster than the second one since action $a$ occurs earlier in the run and since action $b$ occurs at absolute time two in both runs, measured from the start of each run. Accordingly, we define a second variant of our faster–than preorder, where a time step of the slower process is either simulated immediately by the faster one or might be performed later on. As a key result we prove that both variants coincide with our faster–than preorder. Subsequently, this paper develops the preorder's semantic theory: we characterize the coarsest precongruence contained in it, demonstrate that TACS with this precongruence is a conservative extension of CCS with bisimulation, and axiomatize our precongruence for finite sequential processes. We also study the corresponding weak faster–than preorder which abstracts from internal computation. All proofs can be found in a technical report [14].

## 2 Timed Asynchronous Communicating Systems

The novel process algebra TACS conservatively extends CCS [15] by a concept of global, discrete time. This concept is introduced to CCS by including the clock prefixing operator "$\sigma.$" [11] with a non–standard interpretation: a process $\sigma.P$ *can at most* delay one time unit before having to execute process $P$, provided that $P$ can engage in a communication with the environment or in some internal computation. The semantics of TACS is based on a notion of transition system that involves two kinds of transitions, *action transitions* and *clock transitions*. Action transitions, like in CCS, are local handshake communications in which two processes may synchronize to take a joint state change together. A clock represents the progress of time which manifests itself in a recurrent global synchronization event, the clock transition. As indicated before, action and clock transitions are not orthogonal concepts, since time can only pass if the process under consideration cannot engage in an urgent internal computation.

**Syntax.** Let $\Lambda$ be a countable set of actions not including the distinguished unobservable, *internal* action $\tau$. With every $a \in \Lambda$ we associate a *complementary action* $\overline{a}$. We define $\overline{\Lambda} =_{\rm df} \{\overline{a} \,|\, a \in \Lambda\}$ and take $\mathcal{A}$ to denote the set $\Lambda \cup \overline{\Lambda} \cup \{\tau\}$. Complementation is lifted to $\Lambda \cup \overline{\Lambda}$ by defining $\overline{\overline{a}} =_{\rm df} a$. As in CCS [15], an action $a$ communicates with its complement $\overline{a}$ to produce the internal action $\tau$. We let $a, b, \ldots$ range over $\Lambda \cup \overline{\Lambda}$ and $\alpha, \beta, \ldots$ over $\mathcal{A}$ and represent (potential) clock ticks by the symbol $\sigma$. The syntax of TACS is then defined as follows:

$$P \;::=\; \mathbf{0} \;\mid\; x \;\mid\; \alpha.P \;\mid\; \sigma.P \;\mid\; P + P \;\mid\; P|P \;\mid\; P \setminus L \;\mid\; P[f] \;\mid\; \mu x.P$$

where $x$ is a *variable* taken from a countably infinite set $\mathcal{V}$ of variables, $L \subseteq \mathcal{A} \setminus \{\tau\}$ is a *restriction set*, and $f : \mathcal{A} \to \mathcal{A}$ is a *finite relabeling*. A finite relabeling satisfies the properties $f(\tau) = \tau$, $f(\overline{a}) = \overline{f(a)}$, and $|\{\alpha \,|\, f(\alpha) \neq \alpha\}| < \infty$. The set of all terms is abbreviated by $\widehat{\mathcal{P}}$, and we define $\overline{L} =_{\rm df} \{\overline{a} \,|\, a \in L\}$. Moreover, we use the standard definitions for the semantic *sort* $\mathsf{sort}(P) \subseteq \Lambda \cup \overline{\Lambda}$ of some term $P$, *open* and *closed* terms, and *contexts* (terms with a "hole"). A variable is called *guarded* in a term if each occurrence of the variable is within the scope of an action prefix. Moreover, we require for terms of the form $\mu x.P$ that $x$ is guarded in $P$. We refer to closed and guarded terms as *processes*, with the set of all processes written as $\mathcal{P}$.

**Semantics.** The *operational semantics* of a TACS term $P \in \widehat{\mathcal{P}}$ is given by a labeled transition system $\langle \widehat{\mathcal{P}}, \mathcal{A} \cup \{\sigma\}, \longrightarrow, P \rangle$, where $\widehat{\mathcal{P}}$ is the set of states, $\mathcal{A} \cup \{\sigma\}$ the alphabet, $\longrightarrow \subseteq \widehat{\mathcal{P}} \times \mathcal{A} \cup \{\sigma\} \times \widehat{\mathcal{P}}$ the transition relation, and $P$ the start state. Before we proceed, it is convenient to introduce sets $\mathcal{U}(P)$, for all terms $P \in \widehat{\mathcal{P}}$, which include the *urgent actions* in which $P$ can initially engage, as discussed in the introduction. These sets are inductively defined along the structure of $P$, as shown in Table 1. Strictly speaking, $\mathcal{U}(P)$ does not necessarily contain *all* urgent actions. For example, for $P = \tau.0 + \sigma.a.\mathbf{0}$ we have $\mathcal{U}(P) = \{\tau\}$, although action $a$ is semantically also urgent, because the clock transition of $P$ is preempted according to our notion of maximal progress. However, in the sequel

we need the urgent action set of $P$ only for determining whether $P$ can initially perform an urgent $\tau$. For this purpose, our syntactic definition of urgent action sets suffices since $\tau \in \mathcal{U}(P)$ *if and only if* $\tau$ *is semantically urgent in* $P$.

**Table 1.** Urgent action sets

| | | | | | |
|---|---|---|---|---|---|
| $\mathcal{U}(\sigma.P)$ | $=_{\mathrm{df}} \emptyset$ | $\mathcal{U}(\mathbf{0}) = \mathcal{U}(x)$ | $=_{\mathrm{df}} \emptyset$ | $\mathcal{U}(P \setminus L)$ | $=_{\mathrm{df}} \mathcal{U}(P) \setminus (L \cup \overline{L})$ |
| $\mathcal{U}(\alpha.P)$ | $=_{\mathrm{df}} \{\alpha\}$ | $\mathcal{U}(P+Q)$ | $=_{\mathrm{df}} \mathcal{U}(P) \cup \mathcal{U}(Q)$ | $\mathcal{U}(P[f])$ | $=_{\mathrm{df}} \{f(\alpha) \mid \alpha \in \mathcal{U}(P)\}$ |
| $\mathcal{U}(\mu x.P)$ | $=_{\mathrm{df}} \mathcal{U}(P)$ | $\mathcal{U}(P\|Q)$ | $=_{\mathrm{df}} \mathcal{U}(P) \cup \mathcal{U}(Q) \cup \{\tau \mid \mathcal{U}(P) \cap \overline{\mathcal{U}(Q)} \neq \emptyset\}$ | | |

Now, the operational semantics for action transitions and clock transitions can be defined via the *structural operational rules* displayed in Tables 2 and 3, respectively. For action transitions, the rules are exactly the same as for CCS, with the exception of our new clock–prefix operator. For clock transitions, our semantics is set up such that, if $\tau \in \mathcal{U}(P)$, then a clock tick $\sigma$ of $P$ is inhibited. For the sake of simplicity, let us write $P \xrightarrow{\gamma} P'$ instead of $\langle P, \gamma, P' \rangle \in \longrightarrow$, for $\gamma \in \mathcal{A} \cup \{\sigma\}$, and say that $P$ *may engage in* $\gamma$ *and thereafter behave like* $P'$. Sometimes it is also convenient to write $P \xrightarrow{\gamma}$ for $\exists P'.\, P \xrightarrow{\gamma} P'$.

**Table 2.** Operational semantics for TACS (action transitions)

$$\text{Act}\ \frac{}{\alpha.P \xrightarrow{\alpha} P} \qquad \text{Pre}\ \frac{P \xrightarrow{\alpha} P'}{\sigma.P \xrightarrow{\alpha} P'} \qquad \text{Rec}\ \frac{P \xrightarrow{\alpha} P'}{\mu x.P \xrightarrow{\alpha} P'[\mu x.P/x]}$$

$$\text{Sum1}\ \frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'} \qquad \text{Sum2}\ \frac{Q \xrightarrow{\alpha} Q'}{P+Q \xrightarrow{\alpha} Q'}$$

$$\text{Com1}\ \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q} \qquad \text{Com2}\ \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'} \qquad \text{Com3}\ \frac{P \xrightarrow{a} P' \quad Q \xrightarrow{\overline{a}} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$$

$$\text{Rel}\ \frac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]} \qquad \text{Res}\ \frac{P \xrightarrow{\alpha} P'}{P \setminus L \xrightarrow{\alpha} P' \setminus L}\ \alpha \notin L \cup \overline{L}$$

The *action–prefix* term $\alpha.P$ may engage in action $\alpha$ and then behave like $P$. If $\alpha \neq \tau$, then it may also *idle*, i.e., engage in a clock transition to itself, as process $\mathbf{0}$ does. The *clock–prefix* term $\sigma.P$ can engage in a clock transition to $P$ and, additionally, it can perform any action transition that $P$ can, since $\sigma$ represents a delay of *at most* one time unit. The *summation operator* $+$ denotes nondeterministic choice such that $P+Q$ may behave like $P$ or $Q$. Time has to proceed equally on both sides of summation, whence $P+Q$ can engage in a clock transition and delay the nondeterministic choice if and only if both $P$ and $Q$ can. Consequently, e.g., process $\sigma.a.\mathbf{0} + \tau.\mathbf{0}$ cannot engage in a clock transition; in particular, $a$ has to occur without delay if it occurs at all. The *restriction opera-*

*tor* $\backslash L$ prohibits the execution of actions in $L \cup \overline{L}$ and, thus, permits the scoping of actions. $P[f]$ behaves exactly as $P$ where actions are renamed by the *relabeling* $f$. The term $P|Q$ stands for the *parallel composition* of $P$ and $Q$ according to an interleaving semantics with synchronized communication on complementary actions, resulting in the internal action $\tau$. Again, time has to proceed equally on both sides of the operator. The side condition ensures that $P|Q$ can only progress on $\sigma$, if it cannot engage in an urgent $\tau$. Finally, $\mu x.\, P$ denotes *recursion*, i.e., $\mu x.\, P$ behaves as a distinguished solution to the equation $x = P$.

**Table 3.** Operational semantics for TACS (clock transitions)

| | | | | | |
|---|---|---|---|---|---|
| tNil | $\dfrac{-}{\mathbf{0} \xrightarrow{\sigma} \mathbf{0}}$ | tRec | $\dfrac{P \xrightarrow{\sigma} P'}{\mu x.P \xrightarrow{\sigma} P'[\mu x.P/x]}$ | tRes | $\dfrac{P \xrightarrow{\sigma} P'}{P \setminus L \xrightarrow{\sigma} P' \setminus L}$ |
| tAct | $\dfrac{-}{a.P \xrightarrow{\sigma} a.P}$ | tSum | $\dfrac{P \xrightarrow{\sigma} P' \quad Q \xrightarrow{\sigma} Q'}{P + Q \xrightarrow{\sigma} P' + Q'}$ | tRel | $\dfrac{P \xrightarrow{\sigma} P'}{P[f] \xrightarrow{\sigma} P'[f]}$ |
| tPre | $\dfrac{-}{\sigma.P \xrightarrow{\sigma} P}$ | tCom | $\dfrac{P \xrightarrow{\sigma} P' \quad Q \xrightarrow{\sigma} Q'}{P\|Q \xrightarrow{\sigma} P'\|Q'}\ \tau \notin \mathcal{U}(P\|Q)$ | | |

The operational semantics for TACS possesses several important properties [11]. First, it is *time–deterministic*, i.e., processes react deterministically to clock ticks, reflecting the intuition that progress of time does not resolve choices. Formally, $P \xrightarrow{\sigma} P'$ and $P \xrightarrow{\sigma} P''$ implies $P' = P''$, for all $P, P', P'' \in \widehat{\mathcal{P}}$. Second, according to our variant of *maximal progress*, $P \xrightarrow{\sigma}$ if and only if $\tau \notin \mathcal{U}(P)$, for all $P \in \widehat{\mathcal{P}}$.

## 3 Design Choices for Faster–than Relations

In the following we define a reference faster–than relation, called *naive faster–than preorder*, which is inspired by Milner's notions of *simulation* and *bisimulation* [15]. Our main objective is to convince the reader that this simple faster–than preorder with its concise definition is not chosen arbitrarily. This is done by showing that it coincides with two other preorders which formalize a notion of faster–than as well and which are possibly more intuitive.

**Definition 1 (Naive faster–than preorder).** A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a *naive faster–than relation* if, for all $\langle P, Q \rangle \in \mathcal{R}$ and $\alpha \in \mathcal{A}$:

1. $P \xrightarrow{\alpha} P'$ implies $\exists Q'.\, Q \xrightarrow{\alpha} Q'$ and $\langle P', Q' \rangle \in \mathcal{R}$.
2. $Q \xrightarrow{\alpha} Q'$ implies $\exists P'.\, P \xrightarrow{\alpha} P'$ and $\langle P', Q' \rangle \in \mathcal{R}$.
3. $P \xrightarrow{\sigma} P'$ implies $\exists Q'.\, Q \xrightarrow{\sigma} Q'$ and $\langle P', Q' \rangle \in \mathcal{R}$.

We write $P \sqsupseteq_{\mathrm{n}} Q$ if $\langle P, Q \rangle \in \mathcal{R}$ for some naive faster–than relation $\mathcal{R}$.

Note that the behavioral relation $\sqsupseteq_{\mathrm{n}}$, as well as all other behavioral relations on processes defined in the sequel, can be extended to open terms by the usual means of closed substitution [15]. It is fairly easy to see that $\sqsupseteq_{\mathrm{n}}$ is a preorder, i.e., it is transitive and reflexive; moreover, $\sqsupseteq_{\mathrm{n}}$ is the largest naive faster–than relation. Intuitively, $P \sqsupseteq_{\mathrm{n}} Q$ holds if $P$ is faster than (or as fast as) $Q$, and if both processes are functionally equivalent (cf. Clauses (1) and (2)). Here, "*$P$ is faster than $Q$*" means the following: if $P$ may let time pass and the environment of $P$ has to wait, then this should also be the case if one considers the slower (or equally fast) process $Q$ instead (cf. Clause (3)). However, if $Q$ lets time pass, then $P$ is not required to match this behavior. Observe that we use bounded delays and, accordingly, are interested in worst–case behavior. Hence, clock transitions of the fast process must be matched, but not those of the slow process; behavior after an unmatched clock transition can just as well occur quickly without the time step, whence it is catered for in Clause (2).

As the naive faster–than preorder is the basis of our approach, it is very important that its definition is intuitively convincing. There are two immediate questions which arise from our definition.

**Question I.** The first question concerns the observation that Clauses (1) and (3) of Def. 1 require that an action or a time step of $P$ must be matched with just this action or time step by $Q$. What if we are less strict? Maybe we should allow the slower process $Q$ to perform some additional time steps when matching the behavior of $P$. This idea is formalized in the following variant of our faster–than preorder. Here, $\xrightarrow{\sigma}{}^{+}$ and $\xrightarrow{\sigma}{}^{*}$ stand for the transitive and the transitive reflexive closure of the clock transition relation $\xrightarrow{\sigma}$, respectively.

**Definition 2 (Delayed faster–than preorder).** A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a *delayed faster–than relation* if, for all $\langle P, Q\rangle \in \mathcal{R}$ and $\alpha \in \mathcal{A}$:

1. $P \xrightarrow{\alpha} P'$ implies $\exists Q'.\, Q \xrightarrow{\sigma}{}^{*} \xrightarrow{\alpha} \xrightarrow{\sigma}{}^{*} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$.
2. $Q \xrightarrow{\alpha} Q'$ implies $\exists P'.\, P \xrightarrow{\alpha} P'$ and $\langle P', Q'\rangle \in \mathcal{R}$.
3. $P \xrightarrow{\sigma} P'$ implies $\exists Q'.\, Q \xrightarrow{\sigma}{}^{+} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$.

We write $P \sqsupseteq_{\mathrm{d}} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some delayed faster–than relation $\mathcal{R}$.

As usual one can derive that $\sqsupseteq_{\mathrm{d}}$ is a preorder and that it is the largest delayed faster–than relation. In the following we will show that both preorders $\sqsupseteq_{\mathrm{n}}$ and $\sqsupseteq_{\mathrm{d}}$ coincide; the proof of this result is based on a syntactic relation $\succ$ on terms.

**Definition 3.** The relation $\succ \subseteq \widehat{\mathcal{P}} \times \widehat{\mathcal{P}}$ is defined as the smallest relation satisfying the following properties, for all $P, P', Q, Q' \in \widehat{\mathcal{P}}$.

| | | | | |
|---|---|---|---|---|
| Always: | (1) | $P \succ P$ | (2) | $P \succ \sigma.P$ |
| If $P' \succ P$ and $Q' \succ Q$: | (3) | $P'\|Q' \succ P\|Q$ | (4) | $P' + Q' \succ P + Q$ |
| | (5) | $P' \setminus L \succ P \setminus L$ | (6) | $P'[f] \succ P[f]$ |
| If $P' \succ P$ and $x$ guarded in $P$: | (7) | $P'[\mu x.\, P/x] \succ \mu x.\, P$ | | |

Note that relation $\succ$ is not transitive and that it is also defined for open terms. Its essential properties are: (a) $P \xrightarrow{\sigma} P'$ implies $P' \succ P$, for any terms $P, P' \in \widehat{\mathcal{P}}$, and (b) $\succ$ satisfies the clauses of Def. 1, also on open terms; hence, $\succ_{|\mathcal{P}\times\mathcal{P}} \subseteq \sqsupseteq_{\mathrm{n}}$. Crucial for this are Clauses (2) and (7) of the above definition. For (a) we clearly must include Clause (2). Additionally, Clause (7) covers the unwinding of recursion; for its motivation consider, e.g., the transition $\mu x.\,\sigma.a.\sigma.b.x \xrightarrow{\sigma} a.\sigma.b.\mu x.\,\sigma.a.\sigma.b.x$.

**Theorem 4 (Coincidence I).** *The preorders $\sqsupseteq_n$ and $\sqsupseteq_d$ coincide.*

**Question II.** We now turn to a second question which might be raised regarding the definition of the naive faster–than preorder $\sqsupseteq_{\mathrm{n}}$. Should one add a fourth clause to the definition of $\sqsupseteq_{\mathrm{n}}$ that permits, but not requires, the faster process $P$ to match a clock transition of the slower process $Q$? More precisely, $P$ might be able to do whatever $Q$ can do after a time step, or $P$ might itself have to perform a time step to match $Q$. Hence, a candidate for a fourth clause is

(4) $\quad Q \xrightarrow{\sigma} Q'$ implies $\langle P, Q'\rangle \in \mathcal{R}$ or $\exists P'.\, P \xrightarrow{\sigma} P'$ and $\langle P', Q'\rangle \in \mathcal{R}$.

Unfortunately, this requirement is not as sensible as it might appear at first sight. Consider the processes $P =_{\mathrm{df}} \sigma^n.a.\mathbf{0} \,|\, a.\mathbf{0} \,|\, \overline{a}.\mathbf{0}$ and $Q =_{\mathrm{df}} \sigma^n.a.\mathbf{0} \,|\, \sigma^n.a.\mathbf{0} \,|\, \overline{a}.\mathbf{0}$, for $n \geq 1$. Obviously, we expect $P$ to be faster than $Q$. However, $Q$ can engage in a clock transition to $Q' =_{\mathrm{df}} \sigma^{n-1}.a.\mathbf{0} \,|\, \sigma^{n-1}.a.\mathbf{0} \,|\, \overline{a}.\mathbf{0}$. According to Clause (4) and since $P \not\xrightarrow{\sigma}$, we would require $P$ to be faster than $Q'$. This conclusion, however, should obviously be deemed wrong according to our intuition of "faster than."

The point of this example is that process $P$, which is in some components faster than $Q$, cannot mimic a clock transition of $Q$ with a matching clock transition. However, since $P$ is equally fast in the other components, it cannot simply leave out the time step. The solution to this situation is to remember within the relation $\mathcal{R}$ how many clock transitions $P$ missed out and, in addition, to allow $P$ to perform these clock transitions later. Thus, the computation $Q \xrightarrow{\sigma}{}^{n} a.\mathbf{0} \,|\, a.\mathbf{0} \,|\, \overline{a}.\mathbf{0} \xrightarrow{a} \mathbf{0} \,|\, a.\mathbf{0} \,|\, \overline{a}.\mathbf{0} \xrightarrow{a} \mathbf{0} \,|\, \mathbf{0} \,|\, \overline{a}.\mathbf{0}$ of $Q$, where we have no clock transitions between the two action transitions labeled by $a$, can be matched by $P$ with the computation $P \xrightarrow{a} \sigma^n.a.\mathbf{0} \,|\, \mathbf{0} \,|\, \overline{a}.\mathbf{0} \xrightarrow{\sigma}{}^{n} a.\mathbf{0} \,|\, \mathbf{0} \,|\, \overline{a}.\mathbf{0} \xrightarrow{a} \mathbf{0} \,|\, \mathbf{0} \,|\, \overline{a}.\mathbf{0}$. This matching is intuitively correct, since the first $a$ occurs faster in the considered trace of $P$ than in the trace of $Q$, while the second $a$ occurs at the same absolute time, measured from the start of each computation.

**Definition 5 (Family of faster–than preorders).** A family $(\mathcal{R}_i)_{i\in\mathbb{N}}$ of relations over $\mathcal{P}$, indexed by natural numbers (including 0), is a *family of indexed–faster–than relations* if, for all $i \in \mathbb{N}$, $\langle P, Q\rangle \in \mathcal{R}_i$, and $\alpha \in \mathcal{A}$:

1. $P \xrightarrow{\alpha} P'$ implies $\exists Q'.\, Q \xrightarrow{\alpha} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}_i$.
2. $Q \xrightarrow{\alpha} Q'$ implies $\exists P'.\, P \xrightarrow{\alpha} P'$ and $\langle P', Q'\rangle \in \mathcal{R}_i$.
3. $P \xrightarrow{\sigma} P'$ implies (a) $\exists Q'.\, Q \xrightarrow{\sigma} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}_i$, or
   (b) $i > 0$ and $\langle P', Q\rangle \in \mathcal{R}_{i-1}$.

4. $Q \xrightarrow{\sigma} Q'$ implies (a) $\exists P'.\, P \xrightarrow{\sigma} P'$ and $\langle P', Q' \rangle \in \mathcal{R}_i$, or
   (b) $\langle P, Q' \rangle \in \mathcal{R}_{i+1}$.

We write $P \sqsupseteq_{\sim_i} Q$ if $\langle P, Q \rangle \in \mathcal{R}_i$ for some family of indexed–faster–than relations $(\mathcal{R}_i)_{i \in \mathbb{N}}$.

Intuitively, $P \sqsupseteq_{\sim_i} Q$ means that process $P$ is faster than process $Q$ provided that $P$ may delay up to $i$ additional clock ticks which $Q$ does not need to match. Observe that there exists a family of largest indexed–faster–than relations, but it is not clear that these relations are transitive. We establish, however, a stronger result by showing that our naive faster–than preorder $\sqsupseteq_{\sim_{\mathrm{n}}}$ coincides with $\sqsupseteq_{\sim_0}$. The proof of this result uses a family of purely syntactic relations $\succ_i$, for $i \in \mathbb{N}$.

**Definition 6.** The relations $\succ_i \subseteq \widehat{\mathcal{P}} \times \widehat{\mathcal{P}}$, for $i \in \mathbb{N}$, are defined as the smallest relations such that, for all $P, P', Q, Q', P_1, \ldots, P_n \in \widehat{\mathcal{P}}$ and $i, j \in \mathbb{N}$:

| | | | |
|---|---|---|---|
| Always: | (1) | $P \succ_i P$ | |
| If $P_1 \succ P_2 \succ \cdots \succ P_n$: | (2a) | $P_1 \succ_i \sigma^j.P_n$ | |
| If $P' \succ_i P$ and $Q' \succ_i Q$: | (2b) | $\sigma.P' \succ_{i+1} P$ | |
| | (3) | $P'\|Q' \succ_i P\|Q$ | (4) $P' + Q' \succ_i P + Q$ |
| | (5) | $P' \setminus L \succ_i P \setminus L$ | (6) $P'[f] \succ_i P[f]$ |
| If $P' \succ_i P$, $x$ guarded in $P$: | (7a) | $P'[\mu x.\, P/x] \succ_i \mu x.\, P$ | |
| If $P' \succ_i P$, $x$ guarded in $P'$: | (7b) | $\mu x.\, P' \succ_i P[\mu x.\, P'/x]$ | |

Our syntactic relations satisfy the following useful properties:

1. $\succ_i \subseteq \succ_{i+1}$, for all $i \in \mathbb{N}$.
2. $\succ \subseteq \succ_0$; in particular, $P \xrightarrow{\sigma} P'$ implies $P' \succ_0 P$, for any $P, P' \in \widehat{\mathcal{P}}$.
3. $P' \succ P$ (whence, $P \xrightarrow{\sigma} P'$) implies $P \succ_i P'$, for all $i > 0$ and any $P, P' \in \widehat{\mathcal{P}}$.

For the proof of the following theorem, a series of further lemmas is needed, which show in particular that the family of relations $\succ_i$ satisfies the conditions of an indexed–faster–than family.

**Theorem 7 (Coincidence II).** *The preorders $\sqsupseteq_{\sim_n}$ and $\sqsupseteq_{\sim_0}$ coincide.*

## 4 Semantic Theory of our Faster–than Relation

A shortcoming of the naive faster–than preorder $\sqsupseteq_{\sim_{\mathrm{n}}}$, as introduced above, is that it is not compositional. As an example, consider the processes $P =_{\mathrm{df}} \sigma.a.\mathbf{0}$ and $Q =_{\mathrm{df}} a.\mathbf{0}$, for which $P \sqsupseteq_{\sim_{\mathrm{n}}} Q$ holds according to Def. 1. Intuitively, however, this should not be the case, as we expect $P = \sigma.Q$ to be strictly slower than $Q$. Technically, if we compose $P$ and $Q$ in parallel with process $R =_{\mathrm{df}} \overline{a}.\mathbf{0}$, then $P|R \xrightarrow{\sigma} a.\mathbf{0}|\overline{a}.\mathbf{0}$, but $Q|R \not\xrightarrow{\sigma}$, since any clock transition of $Q|R$ is preempted due to $\tau \in \mathcal{U}(Q|R)$. Hence, $P|R \not\sqsupseteq_{\sim_{\mathrm{n}}} Q|R$, i.e., $\sqsupseteq_{\sim_{\mathrm{n}}}$ is *not* a precongruence.

The reason for $P$ and $Q$ being equally fast according to $\sqsupseteq_{\sim_{\mathrm{n}}}$ lies in our operational rules: we allow $Q$ to delay arbitrarily since this might be necessary in a context where no communication on $a$ is possible. As $R$ shows, we have to take a refined view once we fix a context. In order to find the largest precongruence contained in $\sqsupseteq_{\sim_{\mathrm{n}}}$ we must take the urgent action sets of processes into account.

**Definition 8 (Strong faster–than precongruence).** A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a *strong faster–than relation* if, for all $\langle P, Q \rangle \in \mathcal{R}$ and $\alpha \in \mathcal{A}$:

1. $P \xrightarrow{\alpha} P'$ implies $\exists Q'.\, Q \xrightarrow{\alpha} Q'$ and $\langle P', Q' \rangle \in \mathcal{R}$.
2. $Q \xrightarrow{\alpha} Q'$ implies $\exists P'.\, P \xrightarrow{\alpha} P'$ and $\langle P', Q' \rangle \in \mathcal{R}$.
3. $P \xrightarrow{\sigma} P'$ implies $\mathcal{U}(Q) \subseteq \mathcal{U}(P)$ and $\exists Q'.\, Q \xrightarrow{\sigma} Q'$ and $\langle P', Q' \rangle \in \mathcal{R}$.

We write $P \sqsupseteq\!\!\!\sim Q$ if $\langle P, Q \rangle \in \mathcal{R}$ for some strong faster–than relation $\mathcal{R}$.

Again, it is easy to see that $\sqsupseteq\!\!\!\sim$ is a preorder, that it is contained in $\sqsupseteq\!\!\!\sim_{\mathrm{n}}$, and that $\sqsupseteq\!\!\!\sim$ is the largest strong faster–than relation. We also have that $P$ is strictly faster than $\sigma.P$, for all $P \in \mathcal{P}$, which is to be expected intuitively.

**Theorem 9 (Full abstraction).** *The preorder $\sqsupseteq\!\!\!\sim$ is the* largest precongruence *contained in $\sqsupseteq\!\!\!\sim_n$.*

We conclude this section by showing that TACS is a conservative extension of CCS. As noted earlier, we can interpret any process not containing a $\sigma$–prefix as CCS process. Moreover, for all TACS processes, we can adopt the equivalence *strong bisimulation* [15], in signs $\sim$, which is defined just as $\sqsupseteq\!\!\!\sim$ when omitting the third clause of Def. 8. Additionally, we denote the process obtained from some process $P \in \mathcal{P}$ when deleting all $\sigma$'s by $\sigma$*–strip*$(P)$.

**Theorem 10 (Conservativity).** *Let $P$, $Q \in \mathcal{P}$.*

1. *Always $P \sqsupseteq\!\!\!\sim Q$ implies $P \sim Q$.*
2. *If $P$ and $Q$ do not contain any $\sigma$–prefixes, then $P \sqsupseteq\!\!\!\sim Q$ if and only if $Q \sqsupseteq\!\!\!\sim P$ if and only if $P \sim Q$.*
3. *Always $P \sim \sigma$–strip$(P)$; furthermore, $P \xrightarrow{\sigma} P'$ implies $P \sim P'$.*

This shows that our strong faster–than preorder refines strong bisimulation. Moreover, if no bounded delays occur in some processes, then these processes run in zero–time, and our strong faster–than preorder coincides with strong bisimulation. That the bounded delays in TACS processes do not influence any "functional" behavior, is demonstrated in the third part of the above result.

**Axiomatization.** Next, we provide a sound and complete axiomatization of our strong faster–than precongruence $\sqsupseteq\!\!\!\sim$ for the class of finite sequential processes. According to standard terminology, a process is called *finite sequential* if it does neither contain any recursion operator nor any parallel operator. Although this class seems to be rather restrictive at first sight, it is simple and rich enough to demonstrate, by studying axioms, how exactly our semantic theory for $\sqsupseteq\!\!\!\sim$ in TACS differs from the one for strong bisimulation in CCS [15].

The axioms for our strong faster–than precongruence are shown in Table 4, where any axiom of the form $t = u$ should be read as two axioms $t \sqsupseteq u$ and $u \sqsupseteq t$. We write $\vdash t \sqsupseteq u$ if $t \sqsupseteq u$ can be derived from the axioms. Axioms (A1)–(A4), (D1)–(D4), and (C1)–(C5) are exactly the ones for strong bisimulation in CCS [15]. Hence, the semantic theory of our calculus is distinguished from the

one for strong bisimulation by the additional Axioms (P1)–(P5). Intuitively, Axiom (P1) reflects our notion of maximal progress or urgency, namely that a process, which can engage in an internal urgent action, cannot delay. Axiom (P2) states that, if an action occurs "urgent" and "non–urgent" in a term, then it is indeed urgent, i.e., the non–urgent occurrence of the action may be transformed into an urgent one. Axiom (P3) is similar in spirit, but cannot be derived from Axiom (P2) and the other axioms. Axiom (P4) is a standard axiom in timed process algebras and testifies to the fact that time is a deterministic concept which does not resolve choices. Finally, Axiom (P5) encodes our elementary intuition of $\sigma$–prefixes and speed within TACS, namely that any process $t$ is faster than process $\sigma.t$ which might delay the execution of $t$ by one clock tick.

**Table 4.** Axiomatization for finite sequential processes

| | | | | | | |
|---|---|---|---|---|---|---|
| (A1) | $t + u = u + t$ | | (D1) | $\mathbf{0}[f] = \mathbf{0}$ | | |
| (A2) | $t + (u + v) = (t + u) + v$ | | (D2) | $(\alpha.t)[f] = f(\alpha).(t[f])$ | | |
| (A3) | $t + t = t$ | | (D3) | $(\sigma.t)[f] = \sigma.(t[f])$ | | |
| (A4) | $t + \mathbf{0} = t$ | | (D4) | $(t + u)[f] = t[f] + u[f]$ | | |
| (P1) | $\sigma.t + \tau.u = t + \tau.u$ | | (C1) | $\mathbf{0} \setminus L = \mathbf{0}$ | | |
| (P2) | $a.t + \sigma.a.u = a.t + a.u$ | | (C2) | $(\alpha.t) \setminus L = \mathbf{0}$ | | $\alpha \in L \cup \overline{L}$ |
| (P3) | $t + \sigma.t = t$ | | (C3) | $(\alpha.t) \setminus L = \alpha.(t \setminus L)$ | | $\alpha \notin L \cup \overline{L}$ |
| (P4) | $\sigma.(t + u) = \sigma.t + \sigma.u$ | | (C4) | $(\sigma.t) \setminus L = \sigma.(t \setminus L)$ | | |
| (P5) | $t \sqsupseteq \sigma.t$ | | (C5) | $(t + u) \setminus L = (t \setminus L) + (u \setminus L)$ | | |

The correctness of our axioms relative to $\sqsupseteq\!\!\!\!\!\sim$ can be established as usual [15]; note that all axioms are sound for arbitrary processes, not only for finite sequential ones. To prove the completeness of our axiomatization for finite sequential processes, we use a fairly involved notion of normal form; see [14] for details.

**Theorem 11 (Correctness & completeness).** *For finite sequential processes $t$ and $u$ we have: $\vdash t \sqsupseteq u$ if and only if $t \sqsupseteq\!\!\!\!\!\sim u$.*

How to extend our axiomatization to cover parallel composition, too, is non–trivial and still an open problem. The difficulty lies in the lack of a suitable expansion law: observe that $\sigma.a.\mathbf{0} \mid \sigma.b.\mathbf{0}$ is strictly faster than $\sigma.a.\sigma.b.\mathbf{0} + \sigma.b.\sigma.a.\mathbf{0}$. However, since $\sigma$ is synchronized, a more sensible expansion law would try to equate $\sigma.a.\mathbf{0} \mid \sigma.b.\mathbf{0}$ with $\sigma.(a.\mathbf{0} \mid b.\mathbf{0})$. But this law does not hold, since the latter process can engage in an $a$–transition to $\mathbf{0} \mid b.\mathbf{0}$ and is therefore strictly faster. Thus, our situation is the same as in Moller and Tofts' paper [17] which also considers a bisimulation–type faster–than relation for asynchronous processes, but which deals with best–case rather than worst–case timing behavior. It turns out that the axioms for the sequential sub–calculus given in [17] are all true in our setting; however, we have the additional Axioms (P1) and (P2) which both are valid since $\sigma$ is just a potential delay that can occur in certain contexts. Note that also Moller and Tofts do not treat parallel composition completely.

**Abstracting from internal computation.** The strong faster–than precongruence requires that two systems have to match each others action transitions exactly, even those labeled with the internal action $\tau$. Instead, one would like to abstract from $\tau$'s and develop a faster–than precongruence from the point of view of an external observer, as in CCS [15].

We start off with the definition of a *naive weak faster–than preorder* which requires us to introduce the following auxiliary notations. For any action $\alpha$ we define $\hat{\alpha} =_{\mathrm{df}} \epsilon$, if $\alpha = \tau$, and $\hat{\alpha} =_{\mathrm{df}} \alpha$, otherwise. Further, we let $\stackrel{\epsilon}{\Longrightarrow} =_{\mathrm{df}} \stackrel{\tau}{\longrightarrow}^*$ and write $P \stackrel{\alpha}{\Longrightarrow} Q$ if there exist $R$ and $S$ such that $P \stackrel{\epsilon}{\Longrightarrow} R \stackrel{\alpha}{\longrightarrow} S \stackrel{\epsilon}{\Longrightarrow} Q$.

**Definition 12 (Naive weak faster–than preorder).** A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a *naive weak faster–than relation* if, for all $\langle P, Q\rangle \in \mathcal{R}$ and $\alpha \in \mathcal{A}$:

1. $P \stackrel{\alpha}{\longrightarrow} P'$ implies $\exists Q'.\, Q \stackrel{\hat{\alpha}}{\Longrightarrow} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$.
2. $Q \stackrel{\alpha}{\longrightarrow} Q'$ implies $\exists P'.\, P \stackrel{\hat{\alpha}}{\Longrightarrow} P'$ and $\langle P', Q'\rangle \in \mathcal{R}$.
3. $P \stackrel{\sigma}{\longrightarrow} P'$ implies $\exists Q', Q'', Q'''.\, Q \stackrel{\epsilon}{\Longrightarrow} Q'' \stackrel{\sigma}{\longrightarrow} Q''' \stackrel{\epsilon}{\Longrightarrow} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$.

We write $P \sqsupseteq_{\approx_{\mathrm{n}}} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some naive weak faster–than relation $\mathcal{R}$.

Since no urgent action sets are considered, it is easy to see that $\sqsupseteq_{\approx_{\mathrm{n}}}$ is not a precongruence (cf. Def. 8).

**Definition 13 (Weak faster–than preorder).** A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a *weak faster–than relation* if, for all $\langle P, Q\rangle \in \mathcal{R}$ and $\alpha \in \mathcal{A}$:

1. $P \stackrel{\alpha}{\longrightarrow} P'$ implies $\exists Q'.\, Q \stackrel{\hat{\alpha}}{\Longrightarrow} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$.
2. $Q \stackrel{\alpha}{\longrightarrow} Q'$ implies $\exists P'.\, P \stackrel{\hat{\alpha}}{\Longrightarrow} P'$ and $\langle P', Q'\rangle \in \mathcal{R}$.
3. $P \stackrel{\sigma}{\longrightarrow} P'$ implies $\exists Q', Q'', Q'''.\, Q \stackrel{\epsilon}{\Longrightarrow} Q'' \stackrel{\sigma}{\longrightarrow} Q''' \stackrel{\epsilon}{\Longrightarrow} Q'$, $\mathcal{U}(Q'') \subseteq \mathcal{U}(P)$, and $\langle P', Q'\rangle \in \mathcal{R}$.

We write $P \sqsupseteq_{\approx} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some weak faster–than relation $\mathcal{R}$.

Hence, $\sqsupseteq_{\approx}$ is the largest weak faster–than relation and also a preorder. However, $\sqsupseteq_{\approx}$ is still not a precongruence for summation, but the summation fix used for other bisimulation–based timed process algebras proves effective for TACS, too.

**Definition 14 (Weak faster–than precongruence).** A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a *weak faster–than precongruence relation* if, for all $\langle P, Q\rangle \in \mathcal{R}$ and $\alpha \in \mathcal{A}$:

1. $P \stackrel{\alpha}{\longrightarrow} P'$ implies $\exists Q'.\, Q \stackrel{\alpha}{\Longrightarrow} Q'$ and $P' \sqsupseteq_{\approx} Q'$.
2. $Q \stackrel{\alpha}{\longrightarrow} Q'$ implies $\exists P'.\, P \stackrel{\alpha}{\Longrightarrow} P'$ and $P' \sqsupseteq_{\approx} Q'$.
3. $P \stackrel{\sigma}{\longrightarrow} P'$ implies $\mathcal{U}(Q) \subseteq \mathcal{U}(P)$, and $\exists Q'.\, Q \stackrel{\sigma}{\longrightarrow} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}$.

We write $P \sqsupseteq_{\cong} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for a weak faster–than precongruence relation $\mathcal{R}$.

**Theorem 15 (Full–abstraction).** *The relation $\sqsupseteq_{\approx}$ is a precongruence for all operators except summation, and it is the largest such one contained in $\sqsupseteq_{\approx_n}$. Moreover, the relation $\sqsupseteq_{\cong}$ is the* largest precongruence *contained in $\sqsupseteq_{\approx}$, and hence the largest one contained in $\sqsupseteq_{\approx_n}$.*

## 5 Example: A 2–place Storage

We demonstrate the utility of TACS by means of a small example dealing with two implementations of a 2–place storage in terms of an array and a buffer, respectively. Both can be defined using some definition of a 1–place buffer, e.g., $B_e =_{\mathrm{df}} \mu x.\sigma.in.\overline{out}.x$, which can alternately engage in communications with the environment on channels *in* and *out* [15]. Observe that we assume a communication on channel *out* to be urgent, while process $B_e$ may autonomously delay a communication on channel *in* by one clock tick. Finally, subscript $e$ of process $B_e$ should indicate that the 1–place buffer is initially empty. On the basis of $B_e$, one may now define a 2–place array 2ARR and a 2–place buffer 2BUF as follows: $2\textsc{arr} =_{\mathrm{df}} B_e \,|\, B_e$ and $2\textsc{buf} =_{\mathrm{df}} (B_e[c/out] \,|\, B_e[c/in]) \setminus \{c\}$. While 2ARR is simply the parallel composition of two 1–place buffers, 2BUF is constructed by sequencing two 1–place buffers, i.e., by taking the output of the first 1–place buffer to be the input of the second one. Intuitively, we expect the array to behave functionally identical to the buffer, i.e., both should alternate between *in* and *out* actions. However, 2ARR should be faster than 2BUF since it can always output some of its contents immediately. In contrast, 2BUF needs to pass any item from the first to the second buffer cell, before it can output the item [15].

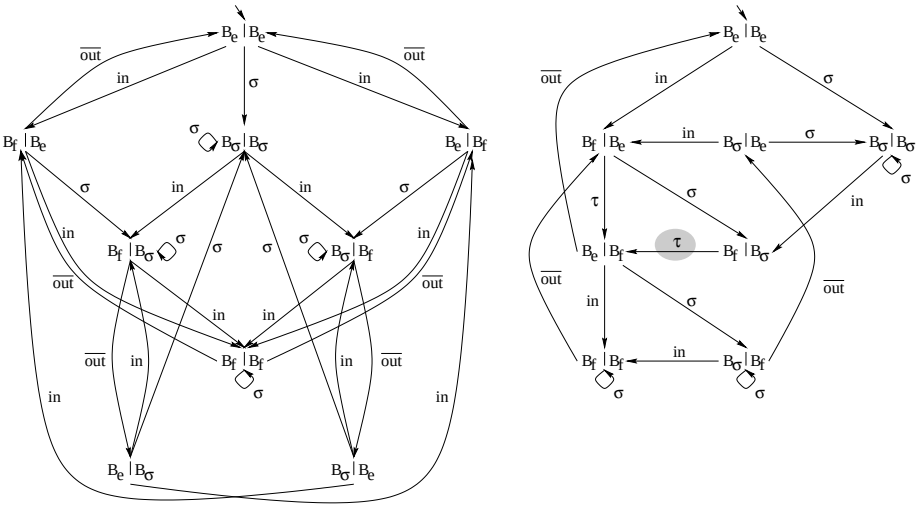


**Fig. 1.** Semantics of the array variant (left) and the buffer variant (right).

The semantics of the 2–place array 2ARR and our 2–place buffer 2BUF are depicted in Fig. 1 on the left and right, respectively. For notational convenience we let $B_\sigma$ stand for the process $\mathrm{in}.\overline{\mathrm{out}}.B_e$ and $B_f$ for $\overline{\mathrm{out}}.B_e$. Moreover, we leave out the restriction operator $\setminus\{c\}$ in the terms depicted for the buffer variant. The highlighted $\tau$–transition indicates an urgent internal step of the buffer.

Hence, process $(\mathrm{B_f}|\mathrm{B}_\sigma) \setminus \{c\}$ cannot engage in a clock transition. The other $\tau$–transition depicted in Fig. 1 is non–urgent. As desired, our semantic theory for TACS relates 2ARR and 2BUF. Formally, this may be witnessed by the weak faster–than relation given in Table 5, whence $2\textsc{arr} \sqsupseteq\!\!\!\approx 2\textsc{buf}$. Moreover, since both 2ARR and 2BUF do not possess any initial internal transitions, they can also easily be proved to be weak faster–than precongruent, according to Def. 14. Thus, $2\textsc{arr} \sqsupseteq\!\!\!\approx 2\textsc{buf}$, i.e., the 2–place array is faster than the 2–place buffer in all contexts, although functionally equivalent, which matches our intuition.

**Table 5.** Pairs in the considered weak faster–than relation

| | | |
|---|---|---|
| $\langle (\mathrm{B_e}\|\mathrm{B_e}) \, , \, (\mathrm{B_e}\|\mathrm{B_e}) \setminus \{c\} \rangle$ | $\langle (\mathrm{B_f}\|\mathrm{B_e}) \, , \, (\mathrm{B_f}\|\mathrm{B_e}) \setminus \{c\} \rangle$ | $\langle (\mathrm{B_e}\|\mathrm{B_f}) \, , \, (\mathrm{B_f}\|\mathrm{B_e}) \setminus \{c\} \rangle$ |
| $\langle (\mathrm{B_f}\|\mathrm{B_e}) \, , \, (\mathrm{B_e}\|\mathrm{B_f}) \setminus \{c\} \rangle$ | $\langle (\mathrm{B_f}\|\mathrm{B}_\sigma) \, , \, (\mathrm{B_f}\|\mathrm{B}_\sigma) \setminus \{c\} \rangle$ | $\langle (\mathrm{B_f}\|\mathrm{B_f}) \, , \, (\mathrm{B_f}\|\mathrm{B_f}) \setminus \{c\} \rangle$ |
| $\langle (\mathrm{B_f}\|\mathrm{B}_\sigma) \, , \, (\mathrm{B}_\sigma\|\mathrm{B_f}) \setminus \{c\} \rangle$ | $\langle (\mathrm{B_e}\|\mathrm{B}_\sigma) \, , \, (\mathrm{B_e}\|\mathrm{B_e}) \setminus \{c\} \rangle$ | $\langle (\mathrm{B}_\sigma\|\mathrm{B_e}) \, , \, (\mathrm{B_e}\|\mathrm{B_e}) \setminus \{c\} \rangle$ |
| $\langle (\mathrm{B}_\sigma\|\mathrm{B_f}) \, , \, (\mathrm{B_e}\|\mathrm{B_f}) \setminus \{c\} \rangle$ | $\langle (\mathrm{B_f}\|\mathrm{B}_\sigma) \, , \, (\mathrm{B_e}\|\mathrm{B_f}) \setminus \{c\} \rangle$ | $\langle (\mathrm{B_e}\|\mathrm{B_f}) \, , \, (\mathrm{B_e}\|\mathrm{B_f}) \setminus \{c\} \rangle$ |
| $\langle (\mathrm{B_f}\|\mathrm{B}_\sigma) \, , \, (\mathrm{B_f}\|\mathrm{B_e}) \setminus \{c\} \rangle$ | $\langle (\mathrm{B}_\sigma\|\mathrm{B_f}) \, , \, (\mathrm{B_f}\|\mathrm{B_e}) \setminus \{c\} \rangle$ | $\langle (\mathrm{B}_\sigma\|\mathrm{B_e}) \, , \, (\mathrm{B}_\sigma\|\mathrm{B_e}) \setminus \{c\} \rangle$ |
| $\langle (\mathrm{B}_\sigma\|\mathrm{B}_\sigma) \, , \, (\mathrm{B}_\sigma\|\mathrm{B}_\sigma) \setminus \{c\} \rangle$ | $\langle (\mathrm{B}_\sigma\|\mathrm{B_f}) \, , \, (\mathrm{B_f}\|\mathrm{B}_\sigma) \setminus \{c\} \rangle$ | $\langle (\mathrm{B}_\sigma\|\mathrm{B_f}) \, , \, (\mathrm{B}_\sigma\|\mathrm{B_f}) \setminus \{c\} \rangle$ |
| $\langle (\mathrm{B_e}\|\mathrm{B}_\sigma) \, , \, (\mathrm{B}_\sigma\|\mathrm{B_e}) \setminus \{c\} \rangle$ | | |

## 6 Discussion and Related Work

The literature includes a large number of papers on timed process algebras [4]. We concentrate only on those which consider faster–than relations.

Research comparing the worst–case timing behavior of asynchronous systems initially centered around DeNicola and Hennessy's testing theory [9]; it was first conducted within the setting of Petri nets [6, 13, 20, 21] and later for a TCSP–style [19] process algebra, called PAFAS [12, 22]. The justification for adopting a testing approach is reflected in a fundamental result stating that the considered faster–than testing preorder based on continuous–time semantics coincides with the analogue testing preorder based on discrete–time semantics [12]. This result depends very much on the testing setting and is different from the sort of discretization obtained for timed automata. In PAFAS, every action has the same integrated upper time bound, namely 1. This gives a more realistic embedding of ordinary process terms, while a CCS–term in TACS runs in zero-time. In contrast, TACS allows one to specify arbitrary upper time bounds easily by nesting $\sigma$–prefixes. Also, the equational laws established for the faster–than testing preorder of PAFAS are quite complicated [22], while the simple axioms presented here provide a clear, comprehensive insight into our semantics.

Regarding other research of faster-than relations, our approach is most closely related to work by Moller and Tofts [17] who developed a bisimulation–based faster–than preorder within the discrete–time process algebra $\ell TCCS$ [16]. In their approach, asynchronous processes are modeled without any progress assumption. Instead, processes may idle arbitrarily long and, in addition, fixed delays may be specified. Hence, their setting is focused on best–case behavior, as the worst–case would be that for an arbitrary long time nothing happens. Moller

and Tofts present an axiomatization of their faster–than preorder for finite sequential processes and discuss the problem of axiomatizing parallel composition, for which only valid laws for special cases are provided. It has to be mentioned here that the axioms and the behavioral preorder of Moller and Tofts do not completely correspond. In fact, writing $\sigma$ for what is actually written (1) in [17], $a.\sigma.b.\mathbf{0} + a.b.\mathbf{0}$ is equally fast as $a.b.\mathbf{0}$, which does not seem to be derivable from the axioms. Also, the intuition behind relating these processes is not so clear, since $a.a.\sigma.b.\mathbf{0} + a.a.b.\mathbf{0}$ is not necessarily faster than or equally fast as $a.a.b.\mathbf{0}$. Since the publication in 1991, also Moller and Tofts noticed this shortcoming of their preorder [*priv. commun.*]. The problem seems to lie in the way in which a transition $P \stackrel{a}{\longrightarrow} P'$ of the faster process is matched: For intuitive reasons, the slower process must be allowed to perform time steps before engaging in $a$. Now the slower process is ahead in time, whence $P'$ should be allowed some additional time steps. What might be wrong is that $P'$ must perform these additional time steps immediately. We assume that a version of our indexed faster–than relation, which relaxes the latter requirement, would be more satisfactory. It would also be interesting to study the resulting preorder and compare it in detail to our faster–than precongruence.

A different idea for relating processes with respect to speed was investigated by Corradini et al. [8] within the so–called *ill–timed–but–well–caused* approach [1, 10]. The key of this approach is that components attach local time stamps to actions; however, actions occur as in an untimed algebra. Hence, in a sequence of actions exhibited by different processes running in parallel, local time stamps might decrease. Due to these "ill–timed" runs, the faster–than preorder of Corradini et al. is difficult to relate to our approach.

Other research compares the efficiency of untimed CCS–like terms by counting internal actions either within a testing framework [7, 18] or a bisimulation–based setting [2, 3]. Except in [7] which does not consider parallel composition, runs of parallel processes are seen to be the interleaved runs of their component processes. Consequently, e.g., $(\tau.a.\mathbf{0} \mid \tau.\overline{a}.b.\mathbf{0}) \setminus \{a\}$ is as efficient as $\tau.\tau.\tau.b.\mathbf{0}$, whereas in our setting $(\sigma.a.\mathbf{0} \mid \sigma.\overline{a}.b.\mathbf{0}) \setminus \{a\}$ is strictly faster than $\sigma.\sigma.\tau.b.\mathbf{0}$.

## 7 Conclusions and Future Work

To consider the worst–case efficiency of asynchronous processes, i.e., those processes whose functional behavior is not influenced by timing issues, we defined the process algebra TACS. This algebra conservatively extends CCS by a clock prefix which represents a delay of at most one time unit, and it takes time to be discrete. For TACS processes we then introduced a simple (bi)simulation–based faster–than preorder and showed this to coincide with two other variants of the preorder, both of which might be intuitively more convincing but which are certainly more complicated. We also developed a semantic theory for our preorder, including a coarsest precongruence result and an axiomatization for finite sequential processes, and investigated a corresponding "weak" preorder.

Regarding future work, we intend to extend our axiomatization to larger classes of processes and also to our weak faster–than preorder, as well as to implement TACS in an automated verification tool.

**Acknowledgments.** We would like to thank the anonymous referees for their valuable comments and suggestions.

## References

[1] L. Aceto and D. Murphy. Timing and causality in process algebra. *Acta Inform.*, 33(4):317–350, 1996.

[2] S. Arun-Kumar and M. Hennessy. An efficiency preorder for processes. *Acta Inform.*, 29(8):737–760, 1992.

[3] S. Arun-Kumar and V. Natarajan. Conformance: A precongruence close to bisimilarity. In *STRICT '95*, Workshops in Comp., pp. 55–68. Springer-Verlag, 1995.

[4] J.C.M. Baeten and C.A. Middelburg. *Process Algebra with Timing: Real Time and Discrete Time*, ch. 10. In Bergstra et al. [5], 2001.

[5] J.A. Bergstra, A. Ponse, and S.A. Smolka, eds. *Handbook of Process Algebra*. Elsevier Science, 2001.

[6] E. Bihler and W. Vogler. Efficiency of token-passing MUTEX-solutions. In *ICATPN '98*, vol. 1420 of *LNCS*, pp. 185–204. Springer-Verlag, 1998.

[7] R. Cleaveland and A. Zwarico. A theory of testing for real time. In *LICS '91*, pp. 110–119. IEEE Computer Society Press, 1991.

[8] F. Corradini, R. Gorrieri, and M. Roccetti. Performance preorder and competitive equivalence. *Acta Inform.*, 34(11):805–835, 1997.

[9] R. DeNicola and M.C.B. Hennessy. Testing equivalences for processes. *TCS*, 34:83–133, 1983.

[10] R. Gorrieri, M. Roccetti, and E. Stancampiano. A theory of processes with durational actions. *TCS*, 140(1):73–94, 1995.

[11] M. Hennessy and T. Regan. A process algebra for timed systems. *Inform. and Comp.*, 117:221–239, 1995.

[12] L. Jenner and W. Vogler. Comparing the efficiency of asynchronous systems. In *ARTS '99*, vol. 1601 of *LNCS*, pp. 172–191. Springer-Verlag, 1999.

[13] L. Jenner and W. Vogler. Fast asynchronous systems in dense time. *TCS*, 254:379–422, 2001.

[14] G. Lüttgen and W. Vogler. A faster-than relation for asynchronous processes. Techn. Rep. 2001-2, ICASE, NASA Langley Research Center, USA, 2001.

[15] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[16] F. Moller and C. Tofts. A temporal calculus of communicating systems. In *CONCUR '90*, vol. 458 of *LNCS*, pp. 401–415. Springer-Verlag, 1990.

[17] F. Moller and C. Tofts. Relating processes with respect to speed. In *CONCUR '91*, vol. 527 of *LNCS*, pp. 424–438. Springer-Verlag, 1991.

[18] V. Natarajan and R. Cleaveland. An algebraic theory of process efficiency. In *LICS '96*, pp. 63–72. IEEE Computer Society Press, 1996.

[19] S. Schneider. An operational semantics for timed CSP. *Inform. and Comp.*, 116(2):193–213, 1995.

[20] W. Vogler. Faster asynchronous systems. In *CONCUR '95*, vol. 962 of *LNCS*, pp. 299–312. Springer-Verlag, 1995.

[21] W. Vogler. Efficiency of asynchronous systems and read arcs in Petri nets. In *ICALP '97*, vol. 1256 of *LNCS*, pp. 538–548. Springer-Verlag, 1997.

[22] W. Vogler and L. Jenner. Axiomatizing a fragment of PAFAS. *ENTCS*, 39, 2000.