Bisimulation on Speed: Worst–case Efficiency ⋆

Gerald Lüttgen [a] and Walter Vogler [b,∗]

[a] *Department of Computer Science, The University of Sheffield, 211 Portobello Street, Sheffield S1 4DP, UK.*

[b] *Institut für Informatik, Universität Augsburg, Universitätsstr. 14, D–86135 Augsburg, Germany.*

Abstract

This paper introduces a novel (bi)simulation–based faster–than preorder which relates asynchronous processes, where the relative speeds of system components are indeterminate, with respect to their worst–case timing behavior. The study is conducted for a conservative extension of the process algebra CCS, called TACS, which permits the specification of upper time bounds on action occurrences. TACS complements work in plain process algebras which compares asynchronous processes with respect to their functional reactive behavior only, and in timed process algebras which focus on analyzing synchronous processes.

The most unusual contribution is in showing that the proposed faster–than preorder coincides with several other preorders, two of which consider the absolute times at which actions occur in system runs. The paper also develops the semantic theory of TACS by studying congruence properties, equational laws, and abstractions from internal actions. Two examples, one dealing with mail delivery and one relating two implementations of a simple storage system, testify to the practical utility of the new theory.

Key words: Asynchronous systems, process algebra, worst–case timing behavior, faster–than relation, bisimulation.

⋆ An extended abstract appeared in K.G. Larsen and M. Nielsen, eds., *12th Intl. Conf. on Concurrency Theory* (CONCUR 2001), vol. 2154 of Lecture Notes in Computer Science, Aalborg, Denmark, August 2001, Springer–Verlag.

∗ Corresponding author.

Email addresses: `g.luettgen@dcs.shef.ac.uk` (Gerald Lüttgen), `vogler@informatik.uni-augsburg.de` (Walter Vogler).

1 Introduction

Process algebras [1–5] provide a widely studied framework for reasoning about the behavior of concurrent systems. Early approaches, including Milner's *Calculus of Communicating Systems* (CCS) [5], focused on semantic issues for asynchronous processes where the relative speeds between processes running in parallel are not bounded, i.e., one process may be arbitrarily slower or faster than another. This leads to a simple and mathematically elegant semantic theory that deals with the functional behavior of systems by describing their causal interactions with their environments. To include time as an aspect of system behavior, *timed process algebras* [6–12] were introduced. They usually model synchronous systems where processes running in parallel are under the regime of a global clock and have a fixed speed. [1] A well–known representative of discrete timed process algebras is Hennessy and Regan's Timed Process Language (TPL) [7] 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 [15] which considers a faster–than preorder within a CCS–based setting, where processes have lower time bounds attached to them [8].

In practice, often upper time bounds, determining how long a process may delay its execution, are important to system designers since these can be used to compare the *worst–case timing behavior* of processes; this corresponds to the progress assumption in [16] and can be realized in other formalisms as well, e.g., in timed automata [17] by employing node invariants. The assumption of upper time bounds for *asynchronous* processes, where the relative speeds of system components are indeterminate, is already exploited in distributed algorithms, as shown by Lynch in [18] in the context of I/O automata. From a concurrency–theoretic point of view, the upper–time–bound assumption was investigated by the second author in the setting of Petri nets [19–21] and was based on De Nicola and Hennessy's notion of testing [22], where the derived must–preorder is interpreted as faster–than relation. Recently, these results have been transferred to a process–algebraic setting [23,24] whose semantics, however, is still based on testing. The fundamental ideas of these approaches, which are also advocated in this paper, are particularly applicable to the analysis of those distributed systems whose behavior is dominated by complex interactions with system environments and between system components. Several case studies in the literature involving mutual exclusion protocols [16] and implementations of buffers [25] testify to this point.

[1] Note that we distinguish this form of synchrony from the one employed in synchronous languages, such as in SCCS [13] and Esterel [14], where the notions of clock and time are implicit rather than explicit.

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 "σ.", where σ 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, however, we view the occurrence of actions as instantaneous. This results in a new process calculus extending CCS, to which we refer as *Timed Asynchronous Communicating Systems* (TACS). To make our intuition of upper–bound delays more precise, consider the processes $\sigma.a.\mathbf{0}$ and $a.\mathbf{0}$, where a denotes an action or port as in CCS. While the former process may delay an enabled communication on port a by one time unit, the latter process must engage in the communication. In this sense, action a is *non–urgent* in $\sigma.a.\mathbf{0}$ but *urgent* in $a.\mathbf{0}$. However, if a communication on port a is not enabled, then process $a.\mathbf{0}$ may wait until some communication partner is ready. Technically, we allow $a.P$ to wait in any case; to enforce a communication resulting in the internal action τ, a time step in TACS is preempted by an *urgent* τ, e.g., by a τ resulting from the synchronized occurrence of two matching urgent communication actions. This is similar to timed process algebras employing the *maximal progress assumption* [7,12]; however, in these algebras and in contrast to TACS, any internal computation is considered to be urgent. For TACS we introduce a (bi)simulation–based *faster–than preorder* which exploits the knowledge of 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 which justifies 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 and probably even more important is the question of how exactly the faster process can match a time step and the subsequent behavior of the slower one. In order to illustrate 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 2 in both runs, measured from the start of each run. With this observation in mind 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 main result we prove that both variants and two relations that combine their underlying ideas coincide with our faster–than preorder that has a more elegant and concise definition. This justifies our faster–than preorder as a reference preorder for relating asynchronous processes with respect to their worst–case timing behavior. In addition, this paper develops the semantic theory of the faster–than preorder which fails to be substitutive regarding the operators choice and parallel composition. We first characterize the coarsest precongruence contained in our preorder, demonstrate that TACS with this precongruence is a conservative extension of CCS with bisimulation, and then axiomatize our precongruence for finite sequential processes. We also study the corresponding weak faster–than preorder, which abstracts from internal computation, and its semantic theory. Two examples of applications of the new theory are offered, one dealing with mail delivery and one relating to two implementations of a simple storage system.

The remainder of this paper is organized as follows. The next section presents the process algebra TACS, while Sec. 3 introduces several variants of a faster–than preorder and shows all of them to coincide. Sec. 4 develops the semantic theory of our preorder and its "weak" corresponding version, which is then applied to two examples in Sec. 5. Finally, Secs. 6 and 7 discuss related work and present our conclusions, respectively. The appendix contains proofs or proof sketches of some auxiliary statements, which are omitted in the main body of the paper.

2 Timed Asynchronous Communicating Systems

This section defines the syntax and semantics of our novel process algebra *Timed Asynchronous Communicating Systems* (TACS) which conservatively extends Milner's CCS [5] by a concept of global, discrete time. This concept is introduced by a non–standard interpretation of clock prefixing "σ." as mentioned in the introduction. Intuitively, a process $\sigma.P$ *can* delay *at most* one time unit before behaving like 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 offers for local handshake communications in which two processes may synchronize to take a joint state change together. In our view, the progress of time manifests itself in a recurrent global synchronization event, the clock transition. As indicated in the introduction, action and clock transitions are not orthogonal concepts since a clock transition can only occur if the process under consideration cannot engage in an urgent internal computation.

Syntax of TACS. Let Λ be a countable[2] set of actions, or ports, not including the distinguished unobservable, *internal* action τ. With every $a \in \Lambda$ we associate a *complementary action* $\overline{a}$. We define $\overline{\Lambda} =_{\mathrm{df}} \{\overline{a} \mid a \in \Lambda\}$ and take $\mathcal{A}$

[2] Most of our results are also valid for finite action sets. However, for our coarsest–precongruence results we must always be able to find "fresh" actions.

Table 1
Urgent action sets

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

to denote the set $\Lambda \cup \overline{\Lambda} \cup \{\tau\}$ of all actions. Complementation is lifted to $\Lambda \cup \overline{\Lambda}$ by defining $\overline{\overline{a}} =_{\rm df} a$. As in CCS [5], an action a communicates with its complement $\overline{a}$ to produce the internal action τ. We let $a, b, \ldots$ range over $\Lambda \cup \overline{\Lambda}$ and $\alpha, \beta, \ldots$ over $\mathcal{A}$ and, moreover, we represent (potential) clock ticks by the symbol σ. The syntax of our language 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 finite *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, for convenience, we define $\overline{L} =_{\rm df} \{\overline{a} \,|\, a \in L\}$. We use the standard definitions for *free* and *bound* variables (where μx binds x), *open* and *closed* terms, and *contexts* (terms with one occurrence of a "hole"). $P[Q/x]$ stands for the term that results when substituting every free occurrence of x in P by Q. A variable is called *guarded* in a term if each occurrence of the variable is in the scope of an action prefix. We require for terms of the form $\mu x.P$ that x is guarded in P. Closed, guarded terms are referred to as *processes*, with the set of all processes written as $\mathcal{P}$, and syntactic equality is denoted by $\equiv$.

Semantics of TACS. 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 indicated in the introduction, the urgent actions are exactly those initial actions that are *not* in the scope of a σ–prefix, e.g., α is urgent in $\alpha.P$. We inductively define $\mathcal{U}(P)$ along the structure of P, as shown in Table 1. Observe that if τ arises from a communication of visible actions a and $\overline{a}$, then it is urgent if so are a and $\overline{a}$.

Now, the operational semantics for action transitions and clock transitions can be defined via *structural operational rules* which are displayed in Tables 2 and 3, respectively. For action transitions, the rules are exactly the same as for CCS, with the exception of the rule for our new clock–prefix operator and the rule for recursion. The latter rule is however equivalent to the standard CCS rule [26]. For clock transitions, our semantics is set up in such a way

Table 2
Operational semantics for TACS (action transitions)

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

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

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

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

Table 3
Operational semantics for TACS (clock transitions)

$$\text{tNil}\ \frac{-}{\mathbf{0} \xrightarrow{\sigma} \mathbf{0}} \qquad \text{tRec}\ \frac{P \xrightarrow{\sigma} P'}{\mu x.P \xrightarrow{\sigma} P'[\mu x.P/x]} \qquad \text{tRes}\ \frac{P \xrightarrow{\sigma} P'}{P \setminus L \xrightarrow{\sigma} P' \setminus L}$$

$$\text{tAct}\ \frac{-}{a.P \xrightarrow{\sigma} a.P} \qquad \text{tSum}\ \frac{P \xrightarrow{\sigma} P' \quad Q \xrightarrow{\sigma} Q'}{P+Q \xrightarrow{\sigma} P'+Q'} \qquad \text{tRel}\ \frac{P \xrightarrow{\sigma} P'}{P[f] \xrightarrow{\sigma} P'[f]}$$

$$\text{tPre}\ \frac{-}{\sigma.P \xrightarrow{\sigma} P} \qquad \text{tCom}\ \frac{P \xrightarrow{\sigma} P' \quad Q \xrightarrow{\sigma} Q'}{P|Q \xrightarrow{\sigma} P'|Q'}\ \tau \notin \mathcal{U}(P|Q)$$

that, if $\tau \in \mathcal{U}(P)$, then a clock tick σ of P is inhibited, in accordance with our adapted variant of maximal progress. 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* γ *and thereafter behave like* P'. Sometimes it is also convenient to write $P \xrightarrow{\gamma}$ for $\exists P'.\, P \xrightarrow{\gamma} P'$.

According to our operational rules, the *action–prefix* term $\alpha.P$ may engage in action α 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 engage in, since σ 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. As a consequence, e.g., process $\sigma.a.\mathbf{0} + \tau.\mathbf{0}$ cannot engage in a clock transition; in particular, a is not urgent, but nevertheless it has to occur without delay if it occurs at all. The *restriction operator* $\setminus L$ 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 τ. Again, time has to proceed equally on both sides of the operator. The side condition ensures that $P|Q$ can only progress on σ, if it cannot engage in any urgent internal computation, in accordance with our notion of maximal progress. Note that predicates within structural operational rules, such as the predicate $\tau \notin \mathcal{U}(P|Q)$ in Rule (tCom), are well understood; see [27] for details on rule formats that treat predicates explicitly and the congruences they imply. Finally, $\mu x.\, P$ denotes *recursion*, i.e., the term $\mu x.\, P$ is a solution to the equation $x = P$.

The operational semantics for TACS possesses several important properties, in analogy to many timed process algebras [7,12]. 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 $P' \equiv P''$, for all $P, P', P'' \in \widehat{\mathcal{P}}$. Second, by our variant of *maximal progress*, a guarded term P can engage in a clock transition exactly if it cannot engage in an urgent internal transition. Formally, $P \xrightarrow{\sigma}$ if and only if $\tau \notin \mathcal{U}(P)$, for all guarded terms P. Third, the interplay between action transitions and clock transitions can be made precise as follows.

Lemma 1 *Let P, P', P'' be processes, with no occurrence of parallel composition in P, and let $\alpha \in \mathcal{A}$.*

(1) $P \xrightarrow{\sigma} P' \xrightarrow{\alpha} P''$ implies $P \xrightarrow{\alpha} P''$.
(2) $P \xrightarrow{\sigma} P'$ and $P \xrightarrow{\alpha} P''$ implies $P' \xrightarrow{\alpha} P''$.

As with the properties of time determinism and maximal progress, the lemma can be proved by induction on the structure of P. Part (1) of Lemma 1 highlights the nature of upper time bounds in TACS, while Part (2) is the persistence property employed, e.g., in TCCS [28]. Note that both statements are invalid for processes involving parallel composition; as an example, consider the process $P \equiv \sigma.a.\mathbf{0} \,|\, \sigma.b.\mathbf{0}$ and action $\alpha \equiv a$.

We conclude this section by two simple lemmas which will be used in the next sections. The first one highlights the interplay between our transition relation and substitution.

Lemma 2 *Let $P, P', Q \in \widehat{\mathcal{P}}$ and $\gamma \in \mathcal{A} \cup \{\sigma\}$.*

(1) $P \xrightarrow{\gamma} P'$ implies $P[\mu x.Q/x] \xrightarrow{\gamma} P'[\mu x.Q/x]$.
(2) x guarded in P and $P[\mu x.Q/x] \xrightarrow{\gamma} P'[\mu x.Q/x]$ implies

$$\exists P'' \in \widehat{\mathcal{P}}.\, P \xrightarrow{\gamma} P'' \text{ and } P'[\mu x.Q/x] \equiv P''[\mu x.Q/x].$$

The second lemma concerns the *sort* of a term P, which is the set of labels of all transitions reachable in the transition system with start state P, i.e., $\mathsf{sort}(P) =_{\mathrm{df}} \{\alpha \in \mathcal{A} \,|\, \exists P'.\, P \longrightarrow^* P' \xrightarrow{\alpha}\}$, where $\longrightarrow^*$ denotes the reflexive and transitive closure of $\longrightarrow$ (when abstracting from transition labels).

Lemma 3 *The set $\mathsf{sort}(P)$ of any term $P \in \widehat{\mathcal{P}}$ is finite.*

This statement follows from the facts that terms have finite length and that relabelings f satisfy the condition $|\{\alpha \,|\, f(\alpha) \neq \alpha\}| < \infty$. A more detailed justification can be given along the lines of the proof for a corresponding statement for PAFAS [23]. The above lemma establishes the well–definedness of some terms constructed below, as TACS just provides a binary summation operator, i.e., only finite summations can be expressed.

3 Design Choices

In the following we define a reference faster–than relation, called *naive faster–than preorder*, which is inspired by Milner's notion of *simulation* [29] and Park's notion of *bisimulation* [30]. 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 several other preorders which formalize a notion of faster–than as well and which are possibly more intuitive. The semantic theory of our faster–than relation will then be developed in the next section.

Definition 4 (Naive faster–than preorder)
A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a naive faster–than relation *if the following conditions hold 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 $P \gtrsim_{nv} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some naive faster–than relation $\mathcal{R}$.

Note that the behavioral relation $\gtrsim_{nv}$, as well as all other behavioral relations on processes defined in the sequel, can be extended to open terms by the means of closed substitution [5], i.e., $P \gtrsim_{nv} Q$ if $P[\vec{R}/\vec{x}] \gtrsim_{nv} Q[\vec{R}/\vec{x}]$, for terms P, Q with free variables in $\vec{x} = (x_1, \dots, x_n)$ and processes $\vec{R} = (R_1, \dots, R_n)$. It is fairly easy to see that $\sqsupseteq_{nv}$ is a preorder, i.e., it is transitive and reflexive; moreover, $\sqsupseteq_{nv}$ is the largest naive faster–than relation. Technically speaking, the naive faster–than preorder refines bisimulation on action transitions by requiring simple simulation on clock transitions. Intuitively, $P \sqsupseteq_{nv} Q$ holds if P is faster than (or at least 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. Intuitively, we use bounded delays and are accordingly 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). We come back to this issue shortly.

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 and with which we are dealing separately in the following sections.

3.1 Question I

The first question emerges from the observation that Clauses (1) and (3) of Def. 4 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 definition of our first variant of the faster–than preorder, which we refer to as *delayed 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 5 (Delayed faster–than preorder)
A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a delayed faster–than relation *if the following conditions hold 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 $P \sqsupseteq_{dly} Q$ if $\langle P, Q\rangle \in \mathcal{R}$ for some delayed faster–than relation $\mathcal{R}$.

As usual, one can derive that $\gtrsim_{dly}$ is a preorder and that it is the largest delayed faster–than relation. In the following we will show that both preorders $\gtrsim_{nv}$ and $\gtrsim_{dly}$ coincide. The proof of this first coincidence result is based on a syntactic relation $\succ$ on terms, which is defined next and which is similar to the progress preorder used in [23]. The objective of its definition is to provide a useful technical handle on the relation between clock transitions and speed; it is constructed such that property

$$P \xrightarrow{\sigma} P' \text{ implies } P' \succ P\,, \tag{1}$$

holds for any $P, P' \in \widehat{\mathcal{P}}$ (cf. Prop. 9(1)).

Definition 6 *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 $P' \succ P$, $Q' \succ Q$:	(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 $P' \succ P$, 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 not only defined for processes but for arbitrary, open terms. The crucial clauses of the above definition are Clauses (2) and (7). Since we want $P \xrightarrow{\sigma} P'$ to imply $P' \succ P$, we clearly must include Clause (2). Additionally, Clause (7) covers the unwinding of recursion; for its motivation consider, e.g., the transition $\mu x.\, \sigma.a.\sigma.b.x \xrightarrow{\sigma} a.\sigma.b.\mu x.\, \sigma.a.\sigma.b.x$.

To establish the desired Property (1) of $\succ$ we need to state and prove some technical lemmas. The first lemma is concerned with the preservation of $\succ$ under substitution as well as with the preservation of substitution by $\succ$.

Lemma 7 (Preservation results)

(1) Let $P, P' \in \widehat{\mathcal{P}}$ such that $P' \succ P$, and let $y \in \mathcal{V}$. Then, y is guarded in P if and only if y is guarded in P'.
(2) Let $P, P', Q \in \widehat{\mathcal{P}}$ such that $P' \succ P$, and let $y \in \mathcal{V}$. Then, $P'[Q/y] \succ P[Q/y]$.
(3) Let $P, Q, Q', R \in \widehat{\mathcal{P}}$ and $x \in \mathcal{V}$ guarded in Q' such that $P \succ Q \equiv Q'[\mu x.R/x]$. Then there exists some $P' \in \widehat{\mathcal{P}}$ satisfying $P \equiv P'[\mu x.R/x]$ and $P' \succ Q'$.

Part (3) will be of importance in the following section (cf. Lemma 35). The next lemma relates $\succ$ to our notion of urgent action sets.

Lemma 8 *Let* $P, Q \in \widehat{\mathcal{P}}$.

(1) If x is guarded in P, then $\mathcal{U}(P[Q/x]) = \mathcal{U}(P)$.
(2) If $Q \succ P$, then $\mathcal{U}(Q) \supseteq \mathcal{U}(P)$.

The proof of Part (1) is an easy induction on the structure of P. Part (2) follows by induction on the inference length of $Q \succ P$. Here, one needs to use Part (1) for Case (7) of Def. 6; note that x is guarded in P' by Lemma 7(1).

Now we have established the machinery needed to prove the above Property (1) and, equally important, to prove that $\succ$ is a naive faster–than relation.

Proposition 9

(1) $P \xrightarrow{\sigma} P'$ implies $P' \succ P$, for all terms $P, P' \in \widehat{\mathcal{P}}$.
(2) The relation $\succ$ satisfies the defining clauses of a naive faster–than relation, also on open terms; hence, $\succ$ restricted to processes is a naive faster–than relation, i.e., $\succ_{|\mathcal{P}\times\mathcal{P}} =_{df} \succ \cap\, (\mathcal{P} \times \mathcal{P}) \subseteq \gtrsim_{nv}$

PROOF. The proof of Part (1) is a straightforward induction on the length of inference of $P \xrightarrow{\sigma} P'$. For proving Part (2) we show that, for $P' \succ P$, the three clauses in the definition of $\gtrsim_{nv}$ are satisfied. This is done by induction on the inference length of $P' \succ P$. We only consider the interesting parts for some of the cases of Def. 6.

(2) $\underline{P \succ \sigma.P}$: Our semantics states that $P \xrightarrow{\alpha} P'$ if and only if $\sigma.P \xrightarrow{\alpha} P'$, for some P', thereby implying the first two clauses in Def. 4. If $P \xrightarrow{\sigma} P'$, then $\sigma.P \xrightarrow{\sigma} P$ and $P' \succ P$ by Part (1).

(3) $\underline{P'|Q' \succ P|Q}$: If $P'|Q' \xrightarrow{\alpha} P_1'|Q'$, for some P_1', due to $P' \xrightarrow{\alpha} P_1'$ (cf. Rule (Com1)), then $P \xrightarrow{\alpha} P_1$ with $P_1' \succ P_1$ and $Q' \succ Q$ by the induction hypothesis. Hence, $P|Q \xrightarrow{\alpha} P_1|Q$ and $P_1'|Q' \succ P_1|Q$. The other cases involving Rules (Com2) and (Com3) are similar.

If $P'|Q' \xrightarrow{\sigma} P_1'|Q_1'$, for some processes P_1' and Q_1', due to $P' \xrightarrow{\sigma} P_1'$ and $Q' \xrightarrow{\sigma} Q_1'$ (cf. Rule (tCom)), then $P \xrightarrow{\sigma} P_1$ and $Q \xrightarrow{\sigma} Q_1$ with $P_1' \succ P_1$ and $Q_1' \succ Q_1$ by the induction hypothesis. Using Lemma 8(2) we conclude from $P'|Q' \xrightarrow{\sigma} P_1'|Q_1'$ that $P|Q \xrightarrow{\sigma} P_1|Q_1$ and $P_1'|Q_1' \succ P_1|Q_1$.

(7) $\underline{P'[\mu x.P/x] \succ \mu x.P}$: By Rule (Rec) any α–transition of $\mu x.P$ is of the form $\mu x.P \xrightarrow{\alpha} P_1[\mu x.P/x]$, for some P_1 with $P \xrightarrow{\alpha} P_1$. Then, by the induction hypothesis, $P' \xrightarrow{\alpha} P_1'$ for some P_1' satisfying $P_1' \succ P_1$. Hence, $P'[\mu x.P/x] \xrightarrow{\alpha} P_1'[\mu x.P/x]$ by Lemma 2(1) since x is guarded in P' by Lemma 7(1), and we obtain $P_1'[\mu x.P/x] \succ P_1[\mu x.P/x]$ by Lemma 7(2).

Further, any α–transition of $P'[\mu x.P/x]$ is of the form $P'[\mu x.P/x] \xrightarrow{\alpha} P_1'[\mu x.P/x]$ for some P_1', where $P' \xrightarrow{\alpha} P_1''$ for some $P_1'' \in \widehat{\mathcal{P}}$ such that

$P_1'[\mu x.P/x] \equiv P_1''[\mu x.P/x]$ by Lemma 2(2), since x is guarded in P' by Lemma 7(1). Thus, by the induction hypothesis, $P \xrightarrow{\alpha} P_1$ with $P_1'' \succ P_1$, as well as $\mu x.P \xrightarrow{\alpha} P_1[\mu x.P/x]$ and $P_1'[\mu x.P/x] \equiv P_1''[\mu x.P/x] \succ P_1[\mu x.P/x]$ by Lemma 7(2). The treatment of clock transitions is analogous.

The other parts are easier to prove and, therefore, are omitted. $\Box$

We are now able to state and prove our first main result.

Theorem 10 (Coincidence I) *The preorders $\gtrsim_{nv}$ and $\gtrsim_{dly}$ coincide.*

PROOF. Clearly, any naive faster–than relation is a delayed one, including $\succ_{|\mathcal{P}\times\mathcal{P}}$ according to Prop. 9(2). Thus, it suffices to show that the largest delayed faster–than relation $\gtrsim_{dly}$ is a naive faster–than relation. Hence, consider some arbitrary processes P and Q such that $P \gtrsim_{dly} Q$.

If $P \xrightarrow{\sigma} P'$ for some process P', then $Q \equiv Q_0 \xrightarrow{\sigma} Q_1 \xrightarrow{\sigma} \cdots \xrightarrow{\sigma} Q_n$ and $P' \gtrsim_{dly} Q_n$, for some $n \geq 1$ and some processes $Q_0, Q_1, \ldots, Q_n$. By Prop. 9(1) we get $Q_n \succ \cdots \succ Q_1 \succ Q$. Since $\succ_{|\mathcal{P}\times\mathcal{P}} \subseteq \gtrsim_{dly}$ (see above) and since $\gtrsim_{dly}$ is transitive, we conclude $P' \gtrsim_{dly} Q_1$.

If $P \xrightarrow{\alpha} P'$ for some process P' and some action α, then we have $Q \equiv Q_0 \xrightarrow{\sigma} Q_1 \xrightarrow{\sigma} \cdots \xrightarrow{\sigma} Q_{n-1} \xrightarrow{\alpha} Q_{n-1}' \xrightarrow{\sigma}{}^{*} Q'$ and $P' \gtrsim_{dly} Q'$, for some $n \geq 1$ and some processes $Q_0, Q_1, \ldots, Q_{n-1}, Q_{n-1}', Q'$. Hence, we may conclude $P' \gtrsim_{dly} Q_{n-1}'$ in analogy to the previous case. Since $Q_{n-1} \succ \cdots \succ Q_0$ by Prop. 9(1), we infer by repeated application of Prop. 9(2) $Q_i \xrightarrow{\alpha} Q_i'$, for $0 \leq i \leq n-1$, such that $Q_{n-1}' \succ \cdots \succ Q_0' \equiv Q''$. As above, this implies $P' \gtrsim_{dly} Q''$ and $Q \xrightarrow{\alpha} Q''$.

The case $Q \xrightarrow{\alpha} Q'$, for some process P' and some action α, is obvious. $\Box$

This coincidence result justifies our preference of the simple and technically more elegant naive faster–than preorder $\gtrsim_{nv}$ over the probably more intuitive delayed faster–than preorder $\gtrsim_{dly}$. Nevertheless, $\gtrsim_{dly}$ could in practice be more useful since there exist delayed faster–than relations that are not naive faster–than relations, such as the relation

$$\{\langle \alpha.\mathbf{0}, \sigma^i.\alpha.\sigma^j.\mathbf{0}\rangle, \langle \alpha.\mathbf{0}, \alpha.\sigma^j.\mathbf{0}\rangle, \langle \mathbf{0}, \sigma^j.\mathbf{0}\rangle, \langle \mathbf{0}, \mathbf{0}\rangle\},$$

for any fixed $i > 1$ or $j > 1$, where σ^i stands for the nesting of i clock prefixes. Note that this refers to the relations that define the preorders and not to the preorders themselves.

3.2 Question II

We now turn to a second question which might be raised regarding the definition of the naive faster–than preorder $\sqsupseteq_{nv}$. Should one add a fourth clause to the definition of $\sqsupseteq_{nv}$ 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 in order to match Q. Hence, a candidate for a fourth clause is

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

Unfortunately, this requirement is not as sensible as it might appear at first. Consider the processes $P =_{\text{df}} \sigma^n.a.\mathbf{0} \,|\, a.\mathbf{0} \,|\, \overline{a}.\mathbf{0}$ and $Q =_{\text{df}} \sigma^n.a.\mathbf{0} \,|\, \sigma^n.a.\mathbf{0} \,|\, \overline{a}.\mathbf{0}$, for $n \geq 1$. Obviously, we expect P to be faster than Q. However, Q can engage in a clock transition to $Q' =_{\text{df}} \sigma^{n-1}.a.\mathbf{0} \,|\, \sigma^{n-1}.a.\mathbf{0} \,|\, \overline{a}.\mathbf{0}$. 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 trace of P than in the trace of Q, while the second a occurs at the same absolute time measured from the system start; only the time relative to the first a is greater for P. Observe that this example also testifies to the need to remember arbitrary large numbers of time steps, as $n \geq 1$ is finite but arbitrary. We formalize the above ideas in the definition of the *indexed faster–than preorder.*

Definition 11 (Family of indexed 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. For our purposes, we are mostly interested in relation $\sqsupseteq\!\!\!\sim_0$. Note that there exists a family of largest indexed faster–than relations, but it is not clear that these relations are transitive. For $\sqsupseteq\!\!\!\sim_0$ this follows from the more interesting result stating that our naive faster–than preorder $\sqsupseteq\!\!\!\sim_{nv}$ coincides with $\sqsupseteq\!\!\!\sim_0$. The proof of this result uses a family of purely syntactic relations $\succ_i$, for $i \in \mathbb{N}$, similar to relation $\succ$ in Def. 6.

Definition 12 *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* $i, j \in \mathbb{N}$*:*

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

Observe that Clauses (7a) and (7b) deal with an unwinding of recursion on both sides of $\succ_i$. This is related to our aim to match clock transitions from both sides of $\mathcal{R}_i$. Similarly, we allow the addition of σ on both sides of $\succ_i$ in Clauses (2a) and (2b) and also in more general situations than in Def. 6. The exact form of Clause (2a) is technically motivated; it relies on the fact that, if P_i is faster than P_{i+1} (for $1 \le i < n$) for syntactic reasons, then P_1 is faster than P_n, and even more so if we burden P_n with additional time steps.

Before presenting our main theorem of this section we state two lemmas whose proofs can be found in the appendix.

Lemma 13 *Let* $P, Q, R \in \widehat{\mathcal{P}}$ *such that* $P \succ_i Q$*, and let* $\alpha \in \mathcal{A}$*.*

(1) $R \succ_0 R$*, for all* $R \in \widehat{\mathcal{P}}$.
(2) Let $P, Q \in \widehat{\mathcal{P}}$ *such that* $P \succ_i Q$*, and let* $\alpha \in \mathcal{A}$*. Then,* $P \xrightarrow{\alpha} P'$ *implies* $\exists Q'.\, Q \xrightarrow{\alpha} Q'$ *and* $P' \succ_i Q'$.

(3) *Let $P, Q \in \widehat{\mathcal{P}}$ such that $P \succ_i Q$, and let $\alpha \in \mathcal{A}$. Then, $Q \xrightarrow{\alpha} Q'$ implies $\exists P'.\, P \xrightarrow{\alpha} P'$ and $P' \succ_i Q'$.*
(4) *Let $P, Q \in \widehat{\mathcal{P}}$ such that $P \succ_0 Q$. Then, $\mathcal{U}(P) \supseteq \mathcal{U}(Q)$.*

This lemma states that $\succ_0$ is reflexive and that the relations $\succ_i$ only relate functionally equivalent terms, in the sense of strong bisimulation. Moreover, it builds a bridge between the relation $\succ_0$ and urgent action sets. The next lemma establishes properties similar to those of Clauses (3) and (4) in Def. 11.

Lemma 14 *Let $P \succ_i Q$ for some $P, Q \in \widehat{\mathcal{P}}$.*

(1) *$P \xrightarrow{\sigma} P'$ implies*
- *either: $i = 0$ and $\exists Q'.\, Q \xrightarrow{\sigma} Q'$ and $P' \succ_i Q'$,*
- *or: $i > 0$ and $P' \succ_{i-1} Q$.*

(2) *$Q \xrightarrow{\sigma} Q'$ implies $P \succ_{i+1} Q'$.*

Using the above lemmas we can now prove the following result.

Theorem 15 (Coincidence II) *The preorders $\gtrsim_{nv}$ and $\gtrsim_0$ coincide.*

PROOF. Let $(\mathcal{R}_i)_{i \in \mathbb{N}}$ be a family of indexed faster–than relations. Then, according to Defs. 11 and 4, $\mathcal{R}_0$ is a naive faster–than relation, whence $\gtrsim_0 \subseteq \gtrsim_{nv}$. For the reverse inclusion consider the largest naive faster–than relation $\gtrsim_{nv}$ and define the family $\mathcal{R}_i$, for $i \in \mathbb{N}$, by $P \mathcal{R}_i Q$ if $\exists R.\, P \gtrsim_{nv} R \succ_i Q$, for $P, Q \in \mathcal{P}$. We check that these $\mathcal{R}_i$ satisfy Def. 11. Consider $P \gtrsim_{nv} R \succ_i Q$.

(1) If $P \xrightarrow{\alpha} P'$ for some process P', then $R \xrightarrow{\alpha} R'$ for some process R' with $P' \gtrsim_{nv} R'$ by the definition of $\gtrsim_{nv}$, as well as $Q \xrightarrow{\alpha} Q'$ for some process Q' with $R' \succ_i Q'$ by Lemma 13(2).
(2) The case $Q \xrightarrow{\alpha} Q'$ for some Q' is analogous and uses Lemma 13(3).
(3) If $P \xrightarrow{\sigma} P'$, then $R \xrightarrow{\sigma} R'$ for some R' with $P' \gtrsim_{nv} R'$. Lemma 14(1) shows $Q \xrightarrow{\sigma} Q'$ for some process Q' with $P' \mathcal{R}_0 Q'$, for $i = 0$, and $P' \mathcal{R}_{i-1} Q$, otherwise.
(4) If $Q \xrightarrow{\sigma} Q'$ for some process Q', then $R \succ_{i+1} Q'$ by Lemma 14(2). Thus, $P \mathcal{R}_{i+1} Q'$.

This finishes the proof, since Lemma 13(1) implies $\gtrsim_{nv} \subseteq \mathcal{R}_0 \subseteq \gtrsim_0$. □

3.3 Combining Both Variants

The delayed and indexed faster–than preorders discussed above reflect two different, but orthogonal ideas for varying the definition of our naive faster–than preorder. It is therefore natural to expect that combining the two ideas also yields a preorder identical to the naive faster–than preorder. Indeed, this turns out to be the case. Our formal account begins with the definition of the combined preorder, which we refer to as delayed–indexed faster–than preorder.

Definition 16 (Family of delayed–indexed 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 delayed–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 $\exists Q'.\, Q \xrightarrow{\sigma}^{*} \xrightarrow{\alpha} \xrightarrow{\sigma}^{*} Q'$ 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 $P \sqsupset\!\!\!\sim_{dly,i} Q$ if $\langle P, Q\rangle \in \mathcal{R}_i$ for some family of delayed–indexed faster–than relations $(\mathcal{R}_i)_{i\in\mathbb{N}}$.

Theorem 17 (Coincidence III) *The preorders $\sqsupset\!\!\!\sim_{nv}$ and $\sqsupset\!\!\!\sim_{dly,0}$ coincide.*

PROOF. It is sufficient to show the validity of $\sqsupset\!\!\!\sim_{dly,0} \subseteq \sqsupset\!\!\!\sim_{dly} = \sqsupset\!\!\!\sim_{nv} = \sqsupset\!\!\!\sim_0 \subseteq \sqsupset\!\!\!\sim_{dly,0}$. The first inclusion is a consequence of the fact that any delayed–indexed faster–than relation with index 0 is a delayed faster–than relation according to Def. 5. Similarly, for the second inclusion observe that any indexed faster–than relation is a delayed–indexed faster–than relation according to Def. 16. The two equalities are the statements of Thms. 10 and 15, respectively. □

The next section develops the semantic theory of the faster–than preorder. In particular, it will turn out that our preorder is not a precongruence, and consequently we will characterize the largest precongruence contained in it. In this light, the above coincidence results are very strong since they state that not only the largest precongruences coincide but already the preorders do. However, this is not always the case as turns out when studying yet another variant of our naive faster–than preorder, or the delayed–indexed faster–than preorder, which differs from the others considered so far, although the largest precongruences coincide. At first sight it might be reasonable to expect that replacing Conds. (1) and (3) of Def. 16 by

(1') $P \xrightarrow{\alpha} P'$ implies $\exists Q' \exists k, l \geq 0.\ Q \xrightarrow{\sigma}^{k} \xrightarrow{\alpha} \xrightarrow{\sigma}^{l} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}_{i+k+l}$

(3') $P \xrightarrow{\sigma} P'$ implies (a) $\exists Q' \exists k \geq 0.\ Q \xrightarrow{\sigma}^{k+1} Q'$ and $\langle P', Q'\rangle \in \mathcal{R}_{i+k}$, or

(b) $i > 0$ and $\langle P', Q\rangle \in \mathcal{R}_{i-1}$

respectively, would not alter the delayed–indexed faster–than preorder. Unfortunately, this is not true although it is obvious from Def. 11 that the altered preorder, which we denote by $\underset{\sim}{\sqsupset}_{alt,0}$, includes the indexed faster–than preorder and thus, by Thm. 15, the naive faster–than preorder. However, the reverse inclusion is not valid as one can inspect by studying the following counterexample: $P =_{\rm df} a.\sigma.\tau.\mathbf{0} + a.\tau.\mathbf{0}$ and $Q =_{\rm df} a.\tau.\mathbf{0}$. Then the family $(\mathcal{R}_i)_{i \in \mathbb{N}}$ defined by $\mathcal{R}_0 =_{\rm df} \{\langle P, Q\rangle, \langle \tau.\mathbf{0}, \tau.\mathbf{0}\rangle, \langle \mathbf{0}, \mathbf{0}\rangle\}$, $\mathcal{R}_1 =_{\rm df} \{\langle \sigma.\tau.\mathbf{0}, \tau.\mathbf{0}\rangle\}$, and $\mathcal{R}_i =_{\rm df} \emptyset$, for $i \geq 2$, is a family of delayed–indexed faster–than relations, in the sense of Conds. (1') and (3'), i.e., $P \underset{\sim}{\sqsupset}_{alt,0} Q$. In particular, transition $P \stackrel{a}{\longrightarrow} \sigma.\tau.\mathbf{0}$ is matched by $Q \stackrel{\sigma}{\longrightarrow} \stackrel{a}{\longrightarrow} \tau.\mathbf{0}$ such that $\langle \sigma.\tau.\mathbf{0}, \tau.\mathbf{0}\rangle \in \mathcal{R}_1$ suffices. But $P \not\underset{\sim}{\sqsupset}_{dly,0} Q$ and $P \not\underset{\sim}{\sqsupset}_{nv} Q$, as can easily be verified. Nevertheless, the largest precongruences contained in $\underset{\sim}{\sqsupset}_{nv}$ and $\underset{\sim}{\sqsupset}_{alt,0}$ coincide, as will be shown in the next section.

Summarizing, we hope to have convinced the reader that our naive faster–than preorder is a sensible candidate for a faster–than preorder, as it coincides with several other candidates that seem to be at least equally appealing but are technically not as simple.

4 Semantic Theory of our Faster–than Preorder

This section focuses (i) on developing a fully–abstract precongruence based on our naive faster–than preorder, (ii) on establishing its semantic theory, and (iii) on introducing a corresponding "weak" variant which abstracts from internal, unobservable actions.

4.1 A Fully–abstract Faster–than Preorder

A shortcoming of the naive faster–than preorder $\underset{\sim}{\sqsupset}_{nv}$, as introduced above, is that it is not compositional. As an example, consider the processes $P =_{\rm df} \sigma.a.\mathbf{0}$ and $Q =_{\rm df} a.\mathbf{0}$, for which $P \underset{\sim}{\sqsupset}_{nv} Q$ holds according to Def. 4. Intuitively, however, this should not be the case since we expect $P \equiv \sigma.Q$ to be strictly slower than Q. Technically, if we compose P and Q in parallel with process $R =_{\rm df} \overline{a}.\mathbf{0}$, then $P|R \stackrel{\sigma}{\longrightarrow} a.\mathbf{0}|\overline{a}.\mathbf{0}$, but $Q|R \stackrel{\sigma}{\not\longrightarrow}$, since any clock transition of $Q|R$ is preempted due to $\tau \in \mathcal{U}(Q|R)$. Hence, $P|R \not\underset{\sim}{\sqsupset}_{nv} Q|R$, i.e., $\underset{\sim}{\sqsupset}_{nv}$ is *not* a precongruence.

The reason for P and Q being equally fast according to $\underset{\sim}{\sqsupset}_{nv}$ lies in our SOS–rules: we allow Q to delay arbitrarily since this might be necessary in a context where no communication on a is possible; thus, an additional potential delay as in P makes no difference; in fact, P and Q have exactly the same transitions. As R shows, we have to take a refined view once we fix a context, and the example indicates that, in order to find the largest precongruence contained in $\sqsupseteq_{nv}$, we have to take the urgent action sets of processes into account. The preorder $\sqsupseteq$ which repairs the precongruence defect of $\sqsupseteq_{nv}$ is defined next. According to $\sqsupseteq$ we generally have that P is strictly faster than $\sigma.P$, which is to be expected intuitively.

Definition 18 (Strong faster–than precongruence)
A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a strong faster–than relation *if the following holds 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 $P \sqsupseteq Q$ if $\langle P, Q \rangle \in \mathcal{R}$ for some strong faster–than relation $\mathcal{R}$.

Again, it is easy to see that $\sqsupseteq$ is a preorder, that it is contained in $\sqsupseteq_{nv}$, and that $\sqsupseteq$ is the largest strong faster–than relation. Note that $\succ$, when restricted to processes, is not only a naive, but also a strong faster–than relation according to Lemma 8(2) and Prop. 9(2).

Theorem 19 (Full abstraction)
The preorder $\sqsupseteq$ is the largest precongruence *contained in $\sqsupseteq_{nv}$.*

PROOF. We first need to establish that $\sqsupseteq$ is a precongruence. This can be done in the usual fashion [5]. Indeed, when comparing our technical framework to the bisimulation approach for the timed process algebra CSA developed in [31], which in turn extends CCS, then most cases of the compositionality proof can be easily adapted. One exception is our clock–prefix operator in TACS, for which we need to show that $P \sqsupseteq Q$ implies $\sigma.P \sqsupseteq \sigma.Q$. This is obvious, however, since the initial clock transition of $\sigma.P$ can be matched by the initial clock transition of $\sigma.Q$ and since all action transitions of $\sigma.P$ and $\sigma.Q$ are those of P and Q according to Rule (Pre). In addition, we present the compositionality proof for parallel composition, as it involves the rather unusual side condition regarding urgent action sets. By the definition of $\sqsupseteq$, it suffices to prove that $\mathcal{R} =_{\mathrm{df}} \{\langle P|R, Q|R \rangle \mid P \sqsupseteq Q,\ R \in \mathcal{P}\}$ is a strong faster–than relation. Therefore, let $\langle P|R, Q|R \rangle \in \mathcal{R}$.

- *Action transitions:* The cases $P|R \xrightarrow{\alpha} S$ and $Q|R \xrightarrow{\alpha} S$, for some $\alpha \in \mathcal{A}$ and some $S \in \mathcal{P}$, follows along the lines of the corresponding cases in CCS [5] and, therefore, are omitted here.
- *Clock transitions:* Let $P|R \xrightarrow{\sigma} S$ for some $S \in \mathcal{P}$. According to the only applicable Rule (tCom) we know that (i) $P \xrightarrow{\sigma} P'$ for some $P' \in \mathcal{P}$, (ii) $R \xrightarrow{\sigma} R'$ for some $R' \in \mathcal{P}$, (iii) $\mathcal{U}(P) \cap \overline{\mathcal{U}(R)} = \emptyset$ as well as $\tau \notin \mathcal{U}(P)$ and $\tau \notin \mathcal{U}(R)$, and (iv) $S \equiv P'|R'$. Since $P \sqsupseteq Q$, there exists a process Q' such

that $\mathcal{U}(Q) \subseteq \mathcal{U}(P)$, $Q \xrightarrow{\sigma} Q'$, and $P' \gtrsim Q'$. Therefore, we may conclude $Q|R \xrightarrow{\sigma} Q'|R'$ by Rule (tCom) since $\mathcal{U}(Q) \cap \overline{\mathcal{U}(R)} = \emptyset$, and $\mathcal{U}(Q|R) = \mathcal{U}(Q) \cup \mathcal{U}(R) \subseteq \mathcal{U}(P) \cup \mathcal{U}(R) = \mathcal{U}(P|R)$, by the definition of urgent action sets and the fact that $\tau \notin \mathcal{U}(P)$, $\tau \notin \mathcal{U}(Q)$, and $\tau \notin \mathcal{U}(R)$. Moreover, $\langle P'|R', Q'|R' \rangle \in \mathcal{R}$ holds by the definition of $\mathcal{R}$.

The proof of the compositionality of recursion requires one to introduce a notion of *strong faster–than up to*. This definition and the compositionality proof itself is very similar to the one in CCS regarding strong bisimulation [5].

We are left with establishing that $\gtrsim$ is the *largest* precongruence contained in $\gtrsim_{nv}$. The proof is a slight adaptation of one for CSA in [31]. As it is non–standard, it is worth presenting it in full here. From universal algebra, it is known that the largest precongruence $\gtrsim_{nv}^{c}$ contained in the preorder $\gtrsim_{nv}$ exists, and that $P \gtrsim_{nv}^{c} Q$ if and only if $C[P] \gtrsim_{nv} C[Q]$ for every TACS context $C[x]$, where a TACS context $C[x]$ is a TACS term with one free occurrence of variable x and no free occurrences of other variables. Recall that, for any context $C[x]$, term $C[P]$ is obtained by substituting P for x in $C[x]$ without any α–conversion, i.e., free variables in P might be captured. As $\gtrsim$ is a precongruence contained in $\gtrsim_{nv}$, we have $\gtrsim \;\subseteq\; \gtrsim_{nv}^{c}$, and it remains to show that $P \gtrsim Q$, for some processes $P, Q \in \mathcal{P}$, whenever $C[P] \gtrsim_{nv} C[Q]$, for all TACS contexts $C[x]$. For this it suffices to consider the relation

$$\gtrsim_{aux} =_{\mathrm{df}} \{\langle P, Q\rangle \mid C_{\mathcal{L}}[P] \gtrsim_{nv} C_{\mathcal{L}}[Q] \text{ for some finite } \mathcal{L} \supseteq \mathsf{sort}(P) \cup \mathsf{sort}(Q)\}\,.$$

Here, $C_{\mathcal{L}}[x] =_{\mathrm{df}} x \mid H_{\mathcal{L}}$ and

$$H_{\mathcal{L}} \;=_{\mathrm{df}}\; \mu x.(e.\mathbf{0} + \textstyle\sum\{\tau.(D_L + d_L.x) \mid L \subseteq \overline{\mathcal{L}}\})\,,$$

where D_L is defined as $\sum_{d \in L} d.\mathbf{0}$. Note that $H_{\mathcal{L}}$ is well–defined according to Lemma 3 due to the finiteness of $\mathcal{L}$. The actions e and d_L and their complements are supposed to be "fresh" actions. In this section we do not exploit the presence of the distinguished action e, but we do so when re–using the above context in the proof of Thm. 32. Note that $\gtrsim_{aux}$ is a preorder; while its reflexivity is obvious, transitivity follows from the property that $C_L[P] \gtrsim_{nv} C_L[Q]$ implies $C_{L'}[P] \gtrsim_{nv} C_{L'}[Q]$, for all $L' \supseteq L \supseteq \mathsf{sort}(P) \cup \mathsf{sort}(Q)$. To finish off our proof of Thm. 19, it is sufficient to establish the inclusion $\gtrsim_{aux} \;\subseteq\; \gtrsim$, since the inclusion $\gtrsim_{nv}^{c} \;\subseteq\; \gtrsim_{aux}$ obviously holds.

We show that $\gtrsim_{aux}$ is a strong faster–than relation according to Def. 18. Let P, $Q \in \mathcal{P}$ such that $P \gtrsim_{aux} Q$, i.e., we have $C_{\mathcal{L}}[P] \gtrsim_{nv} C_{\mathcal{L}}[Q]$ for some finite $\mathcal{L} \supseteq \mathsf{sort}(P) \cup \mathsf{sort}(Q)$ by the definition of $\gtrsim_{aux}$. In the following we consider two cases distinguishing whether process P performs an action transition or a clock transition. In each case the transition of P leads to a transition of $C_{\mathcal{L}}[P]$. According to the definition of $\gtrsim_{nv}$, matching transitions must exist which mimic each step. From their existence we may conclude additional conditions which are sufficient to establish $\underset{\sim}{\sqsupset}_{aux}$ as a strong faster–than relation.

- *Situation 1:* Let $P \xrightarrow{\alpha} P'$ for some process P' and some action α. According to our operational semantics we have $C_{\mathcal{L}}[P] \equiv P|H_{\mathcal{L}} \xrightarrow{\alpha} P'|H_{\mathcal{L}} \equiv C_{\mathcal{L}}[P']$. This transition can only be matched by a corresponding transition of Q, say $Q \xrightarrow{\alpha} Q'$ for some Q'. This is even true in case $\alpha \equiv \tau$, because the τ–successors of $H_{\mathcal{L}}$ have the distinguished actions d_L enabled. Therefore, we have $C_{\mathcal{L}}[Q] \equiv Q|H_{\mathcal{L}} \xrightarrow{\alpha} Q'|H_{\mathcal{L}} \equiv C_{\mathcal{L}}[Q']$ and $C_{\mathcal{L}}[P'] \underset{\sim}{\sqsupset}_{nv} C_{\mathcal{L}}[Q']$. Because $\mathsf{sort}(P') \subseteq \mathsf{sort}(P)$ and $\mathsf{sort}(Q') \subseteq \mathsf{sort}(Q)$, we have $\mathcal{L} \supseteq \mathsf{sort}(P') \cup \mathsf{sort}(Q')$, whence $P' \underset{\sim}{\sqsupset}_{aux} Q'$. A transition $Q \xrightarrow{\alpha} Q'$ can be matched analogously.
- *Situation 2:* Let $P \xrightarrow{\sigma} P'$ for some process P'.

$$\begin{array}{ccccc}
P \mid & H_{\mathcal{L}} & \underset{\sim}{\sqsupset}_{nv} & Q \mid & H_{\mathcal{L}} \\
 & \downarrow \tau & & & \downarrow \tau \\
P \mid & (D_L + d_L.H_{\mathcal{L}}) & \underset{\sim}{\sqsupset}_{nv} & Q \mid & (D_L + d_L.H_{\mathcal{L}}) \\
\downarrow \sigma & & & \downarrow \sigma & \\
P' \mid & (D_L + d_L.H_{\mathcal{L}}) & \underset{\sim}{\sqsupset}_{nv} & Q' \mid & (D_L + d_L.H_{\mathcal{L}}) \\
 & \downarrow d_L & & & \downarrow d_L \\
P' \mid & H_{\mathcal{L}} & \underset{\sim}{\sqsupset}_{nv} & Q' \mid & H_{\mathcal{L}}
\end{array}$$

As illustrated in the above figure we let $C_{\mathcal{L}}[P]$ perform a τ–transition to $P|H_L$, where $H_L =_{\mathrm{df}} D_L + d_L.H_{\mathcal{L}}$ and $L =_{\mathrm{df}} \{\overline{c} \,|\, c \in (\mathsf{sort}(P) \cup \mathsf{sort}(Q)) \setminus \mathcal{U}(P)\}$; note that $L \subseteq \overline{\mathcal{L}}$. Then, $P|H_L$ can perform a clock transition to $P'|H_L$ according to Rule (tCom). Finally, we let $P'|H_L$ engage in the d_L–transition to $P'|H_{\mathcal{L}}$. Process $C_{\mathcal{L}}[Q]$ has to match the first step by a τ–transition to $Q|H_L$ since only this term has the distinguished action d_L enabled.

Now we take a closer look at the second step. We have to match a clock transition. Therefore, Q has to perform a clock transition to some Q', and H_L has to idle, i.e., $Q|H_L \xrightarrow{\sigma} Q'|H_L$. According to Rule ($\mathsf{tCom}$), the condition $\mathcal{U}(Q) \cap \overline{\mathcal{U}(H_L)} = \emptyset$ has to be satisfied. Because of the choice of L, this implies $\mathcal{U}(Q) \subseteq \mathcal{U}(P)$.

Finally, the last step can only be matched by the transition $Q'|H_L \xrightarrow{d_L} Q'|H_{\mathcal{L}}$. Thus, $C_{\mathcal{L}}[P'] \equiv P'|H_{\mathcal{L}} \underset{\sim}{\sqsupset}_{nv} Q'|H_{\mathcal{L}} \equiv C_{\mathcal{L}}[Q']$.

Since $\mathsf{sort}(P') \subseteq \mathsf{sort}(P)$ as well as $\mathsf{sort}(Q') \subseteq \mathsf{sort}(Q)$, it follows in analogy to Situation (1) that $P' \underset{\sim}{\sqsupset}_{aux} Q'$.

Thus, $\underset{\sim}{\sqsupset}_{aux}$ is a strong faster–than relation, i.e., $\underset{\sim}{\sqsupset}_{aux} \subseteq \underset{\sim}{\sqsupset}$ according to Def. 18. Hence, $\underset{\sim}{\sqsupset}^c_{nv} \subseteq \underset{\sim}{\sqsupset}_{aux} \subseteq \underset{\sim}{\sqsupset}$ which together with the inclusion $\underset{\sim}{\sqsupset} \subseteq \underset{\sim}{\sqsupset}^c_{nv}$ obtained earlier yields $\underset{\sim}{\sqsupset} = \underset{\sim}{\sqsupset}^c_{nv}$, as desired. □

In Sec. 3 we showed that the naive faster–than preorder coincides with several other variants, in particular the delayed faster–than preorder and the indexed faster–than preorder. This very strong result immediately implies the coincidence of the largest precongruences contained in these preorders. It should be noted that alternative characterizations of the delayed faster–than precongruence and the indexed faster–than precongruence can be given similar in style to the definitions of the corresponding preorders. Since these characterizations are not of importance for the remainder of this paper, they are omitted here.

We are now able to finish our study of alternative definitions of our faster–than preorder from Sec. 3.3 by establishing that the precongruences induced by the naive faster–than preorder and the altered delayed–indexed faster–than preorder coincide.

Theorem 20 (Coincidence IV)
The largest precongruences $\underset{\sim}{\sqsupset} = \underset{\sim}{\sqsupset}^c_{nv}$ and $\underset{\sim}{\sqsupset}^c_{alt,0}$ coincide.

The claim follows by universal algebra from the inclusion chain $\underset{\sim}{\sqsupset}^c_{nv} \subseteq \underset{\sim}{\sqsupset}^c_{alt,0} \subseteq \underset{\sim}{\sqsupset}_{nv}$, which implies $\underset{\sim}{\sqsupset} = \underset{\sim}{\sqsupset}^c_{nv} = \underset{\sim}{\sqsupset}^c_{alt,0}$. The first inclusion of this chain is implied by $\underset{\sim}{\sqsupset}_{nv} = \underset{\sim}{\sqsupset}_0 \subseteq \underset{\sim}{\sqsupset}_{alt,0}$ which immediately follows from Thm. 15 and the definitions of these preorders. The second inclusion $\underset{\sim}{\sqsupset}^c_{alt,0} \subseteq \underset{\sim}{\sqsupset}_{nv}$ is more challenging to establish. We first define an auxiliary relation

$$\begin{aligned}\underset{\sim}{\sqsupset}_{aux'} =_{\mathrm{df}} \{\langle P, Q\rangle \mid\ & (P|K_{\mathcal{L}}) \setminus \mathcal{L} \underset{\sim}{\sqsupset}_{alt,0} (Q|K_{\mathcal{L}}) \setminus \mathcal{L} \\ & \text{for some finite } \mathcal{L} \supseteq \mathsf{sort}(P) \cup \mathsf{sort}(Q)\,\}\,,\end{aligned}$$

where

$$K_{\mathcal{L}} =_{\mathrm{df}} \mu x.\, (h.\mathbf{0} + \tau.(\sigma.\tau.\mathbf{0} + f_\sigma.x) + \sum_{a\in\mathcal{L}} \tau.(\tau.\mathbf{0} + f_{\overline{a}}.\mathbf{0} + \overline{a}.x))\,,$$

and h, f_σ, and $f_{\overline{a}}$, for $a \in \mathcal{L}$, are distinguished actions, i.e., they and their complements are not actions in $\mathcal{L}$. This reduces the proof to establishing $\underset{\sim}{\sqsupset}_{aux'} \subseteq \underset{\sim}{\sqsupset}_{nv}$ which can be done in a fashion similar in style to the largest precongruence part of the proof of Thm. 19; see the appendix for more details.

We conclude this section by showing that TACS is a conservative extension of CCS [5]. Observe that, due to the format of Rules (tNil) and (tAct), TACS is not an *operational* conservative extension in the sense of [32]. Our notion of conservativity relates our strong faster–than precongruence with strong bisimulation in CCS [5]. As noted earlier we can interpret any process not containing a σ–prefix as a CCS process, since then all relevant semantic rules for action transitions are identical to the ones in CCS. Moreover, for all TACS terms, we can adopt the equivalence *strong bisimulation* [5], denoted by $\sim$, which is defined just as $\underset{\sim}{\sqsupset}$ when omitting the third clause of Def. 18. Furthermore, we denote the term obtained from some term $P \in \widehat{\mathcal{P}}$ when deleting all σ's by σ–*strip*(P). We may now state the following conservativity results.

Theorem 21 (Conservativity) *Let* $P, Q \in \mathcal{P}$.

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

PROOF. Part 1 is an immediate consequence of the definitions of $\sim$ and $\underset{\sim}{\sqsupset}$.

Part 2 follows by the fact that terms without σ–prefixes (i) can only engage in a clock transition to themselves, namely if and only if no internal transition is enabled, and (ii) possess the same urgent actions whenever they are related by $\underset{\sim}{\sqsupset}$ or $\sim$, since any action they can perform is urgent. [3]

For the first claim of Part 3, one shows by induction on the structure of process P that the action transitions of σ–*strip*(P) are exactly all transitions σ–*strip*$(P) \xrightarrow{\alpha} \sigma$–*strip*$(P')$ where $P \xrightarrow{\alpha} P'$. For the second claim of the third part, one first proves that $P \xrightarrow{\sigma} P'$ implies that σ–*strip*(P) and σ–*strip*(P') are identical up to unfolding of recursion. Then, one applies the first claim to finish the proof. $\square$

This shows that our strong faster–than preorder refines the well–established notion of 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. In other words, the strong faster–than preorder is thus restricted to considering the "functional" behavior of

[3] Alternatively, this second part may be concluded by inspecting the axiomatization of the strong faster–than precongruence, which can be found in the next section. Note that all axioms, except Axiom (P5) that deals with σ–prefixes, are valid in both directions, "$\sqsupseteq$" and "$\sqsubseteq$".

such processes only, irrespective of their relative speeds. That the bounded delays in TACS processes do not influence any "functional" behavior is demonstrated in the third part of Thm. 21.

Although the above embedding of CCS yields the technical conservation result stated in Thm. 21(2), this might intuitively not be very pleasing: one might expect that the parallel execution of actions is faster than their arbitrary sequential execution, but the result shows that processes $a.\mathbf{0}|b.\mathbf{0}$ and $a.b.\mathbf{0}+b.a.\mathbf{0}$ are equally fast with respect to $\underset{\sim}{\sqsupset}$. Intuitively, for things happening with no time between them, it is difficult to see whether they happened one after the other or together. Of course, the zero–time between a and b is just a mathematical abstraction, but a useful one; it stands for a very short, negligible time. As an alternative, one could follow the approach of [23] and assume that actions might take some time, and for a uniform embedding of CCS one can give each action a bounded delay of one. Technically, this means to embed ordinary CCS–terms into TACS by inserting a σ–prefix before each action. Thm. 21(2) shows that this translation does not change any "functional" behavior. With this embedding, however, the classical expansion law "$a.\mathbf{0} \,|\, b.\mathbf{0} \;=\; a.b.\mathbf{0} + b.a.\mathbf{0}$" is not preserved due to timing: $\sigma.a.\mathbf{0} \,|\, \sigma.b.\mathbf{0}$ is strictly faster than $\sigma.a.\sigma.b.\mathbf{0} + \sigma.b.\sigma.a.\mathbf{0}$; consider the matching of the initial clock transition.

4.2 Axiomatization

In this section we provide a sound and complete axiomatization of our strong faster–than precongruence $\underset{\sim}{\sqsupset}$ 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 $\underset{\sim}{\sqsupset}$ in TACS differs from the one for strong bisimulation in CCS [5]. We refer the reader to the end of this section for a discussion on the implications when considering to axiomatize larger classes of processes. As a notational convention we write $\mathcal{P}^{\mathsf{fin}}_{\mathsf{seq}}$ for the set of all finite sequential processes, ranged over by s, t, and u.

Now we turn to the axioms for strong faster–than precongruence which are displayed 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. The correctness of our axioms with respect to $\underset{\sim}{\sqsupset}$ can be established as usual [5]. Axioms (A1)–(A4), (D1)–(D4), and (C1)–(C5) are exactly the ones for strong bisimulation in CCS. Hence, the semantic theory of our calculus is distinguished from the one for strong bisimulation by the

Table 4
Axioms 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)	$(t+u)\setminus L$	$=$	$(t\setminus L)+(u\setminus L)$	

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. To see this, consider the instance $\mathbf{0}+\sigma.\mathbf{0}=0$, or due to Axioms (A4) and (P5) simply $\sigma.\mathbf{0}\sqsupseteq\mathbf{0}$, and observe in Table 4 that there is no applicable axiom that allows one to ever remove the σ in $\sigma.\mathbf{0}$. Indeed, it would have been sufficient to include $\sigma.\mathbf{0}\sqsupseteq\mathbf{0}$ instead of Axiom (P3), from which $\vdash P+\sigma.P\sqsupseteq P$ follows for finite sequential processes P by induction on the size of P. The motivation for including Axiom (P3) in its present form is due to its soundness for arbitrary TACS processes, not only for finite sequential ones; this is also true of the other axioms. The soundness proof of Axiom (P3) "$\sqsupseteq$" involves establishing that $\{\langle P+Q,P\rangle \,|\, Q\stackrel{\sigma}{\longrightarrow}P\}\cup\{\langle P,P\rangle \,|\, P\in\mathcal{P}\}$ is a strong faster–than relation (cf. Lemma 1(2) and the persistency axiom in [33]). Axiom (P4) is a standard axiom in timed process algebras and testifies to the fact that time is a deterministic concept that does not resolve choices. Finally, Axiom (P5) encodes our elementary intuition of σ–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. Its correctness follows from the facts that $t\succ\sigma.t$ by Def. 6(2) and that $\succ$ is a strong faster–than precongruence by Prop. 9(2), Lemma 8(2), and Def. 18.

To prove the completeness of our axiomatization for finite sequential processes, we introduce a notion of *normal form*, based on the following definition. A finite sequential process t is *in summation form* if it is of the shape

$$t \;\equiv\; \sum_{i \in I} \alpha_i.t_i \;\; [+\, \sigma.t_\sigma\,]$$

where (i) I denotes a finite index set, (ii) all the t_i are in summation form, (iii) the subterm in brackets is optional and, if it exists, t_σ is in summation form, and (iv) $\alpha_i \in \mathcal{A}$, for all $i \in I$. Moreover, $\sum$ is the indexed version of $+$; we adopt the convention that the sum over the empty index set is identified with process $\mathbf{0}$. As expected, we obtain the following result.

Proposition 22 *For any $t \in \mathcal{P}^{\mathit{fin}}_{\mathit{seq}}$, there exists some $u \in \mathcal{P}^{\mathit{fin}}_{\mathit{seq}}$ in summation form such that $\vdash t = u$.*

In the remainder, the following definition of the set of initial actions, in which some process t in summation form can engage in, will prove useful: $\mathcal{I}(t) =_{\mathrm{df}} \mathcal{U}(t)\ [\cup\ \mathcal{I}(t_\sigma)\ ]$. It is easy to establish that $\mathcal{I}(t)$ is compatible with our operational semantics, i.e., the equality $\mathcal{I}(t) = \{\alpha \in \mathcal{A} \,|\, t \xrightarrow{\alpha}\}$ holds.

Definition 23 (Normal form)
The process $\sum_{i \in I} \alpha_i.t_i \;\; [+\, \sigma.t_\sigma\,]$ in summation form is in normal form *if all terms t_i, for $i \in I$, are in normal form and, in case the optional term in brackets is present, the following is satisfied: (i) $t_\sigma \not\equiv \mathbf{0}$; (ii) $\forall i \in I.\ \alpha_i \not\equiv \tau$; (iii) $\forall i \in I.\ \alpha_i \notin \mathcal{I}(t_\sigma)$; and (iv) term t_σ is in normal form.*

Before we state the key proposition that every finite sequential process can be transformed into a process in normal form, we note that Conds. (ii) and (iii) exactly correspond to our abovementioned intuitions regarding Axioms (P1) and (P2), respectively.

Proposition 24 *For any $t \in \mathcal{P}^{\mathit{fin}}_{\mathit{seq}}$, there exists some $u \in \mathcal{P}^{\mathit{fin}}_{\mathit{seq}}$ in normal form such that $\vdash t = u$ and $\mathcal{U}(t) \subseteq \mathcal{U}(u)$.*

Note that the set of urgent actions might increase when transforming a process into normal form due to the application of Axiom (P1), whereas the set of initial actions does not change. This former inclusion is exploited in the completeness proof of our axiomatization. However, before proceeding to our completeness theorem we state a technical lemma.

Lemma 25 *Let $t \equiv \sum_{i \in I} \alpha_i.t_i\ [+\ \sigma.t_\sigma\ ]$ and $u \equiv \sum_{j \in J} \beta_j.u_j\ [+\ \sigma.u_\sigma\ ]$ be processes in normal form such that $t \sqsupseteq\!\!\!\!\raisebox{-0.6ex}{$\sim$}\, u$. Moreover, let $B \subseteq \{\beta_j \,|\, j \in J\}$.*

(1) $\{\beta_j \,|\, j \in J\} \subseteq \{\alpha_i \,|\, i \in I\}$.
(2) $\sum_{\{i \in I \,|\, \alpha_i \in B\}} \alpha_i.t_i \gtrsim \sum_{\{j \in J \,|\, \beta_j \in B\}} \beta_j.u_j$.
(3) $\sum_{\{i \in I \,|\, \alpha_i \notin B\}} \alpha_i.t_i\ [+\ \sigma.t_\sigma\ ] \gtrsim \sum_{\{j \in J \,|\, \beta_j \notin B\}} \beta_j.u_j\ [+\ \sigma.u_\sigma\ ]$.

We are now able to state and prove the main result of this section.

Theorem 26 (Correctness & completeness)
For finite sequential processes t and u we have: $\vdash t \sqsupseteq u$ if and only if $t \gtrsim u$.

PROOF. The correctness "$\Longrightarrow$" of our axiom system follows by induction on the length of the inference $\vdash t \sqsupseteq u$, as usual; we leave it as an exercise to the reader to show that $\sqsupseteq$ may be safely replaced by $\gtrsim$ in each axiom. Thus, we are left with proving completeness "$\Longleftarrow$". By Prop. 24 we may assume that the processes t and u are in normal form with $t \equiv \sum_{i \in I} \alpha_i.t_i \; [+\, \sigma.t_\sigma\,]$ and $u \equiv \sum_{j \in J} \beta_j.u_j \; [+\, \sigma.u_\sigma\,]$. We proceed by induction on the sum of the process sizes of t and u as follows. For the induction base we have $t \equiv u \equiv \mathbf{0}$; hence, $\vdash \mathbf{0} \sqsupseteq \mathbf{0}$ trivially holds and we are left with the induction step.

We first consider the case that neither t nor u possesses an optional σ–summand. According to the definition of $\gtrsim$, there exists for each $i' \in I$ some $j' \in J$ such that $\alpha_{i'} = \beta_{j'}$ and $t_{i'} \gtrsim u_{j'}$. By induction hypothesis we may conclude $\vdash t_{i'} \sqsupseteq u_{j'}$, whence $\vdash \alpha_{i'}.t_{i'} + \sum_{j \in J} \beta_j.u_j \sqsupseteq \beta_{j'}.u_{j'} + \sum_{j \in J} \beta_j.u_j = \sum_{j \in J} \beta_j.u_j$ by Axiom (A3) and possibly Axioms (A1) and (A2). By repeating this reasoning for each $i \in I$, we obtain $\vdash \sum_{i \in I} \alpha_i.t_i + \sum_{j \in J} \beta_j.u_j = t + u \sqsupseteq u = \sum_{j \in J} \beta_j.u_j$. Analogously, we can infer $\vdash t \sqsupseteq t + u$. Hence, $\vdash t \sqsupseteq u$ by transitivity.

Otherwise, we apply Lemma 25(2) to t, u, and $B = \{\beta_j \,|\, j \in J\}$, which yields $\sum_{\{i \in I \,|\, \alpha_i \in B\}} \alpha_i.t_i \gtrsim \sum_{\{j \in J \,|\, \beta_j \in B\}} \beta_j.u_j$. As at least one of t_σ and u_σ is missing when compared to t and u, we may apply the induction hypothesis to conclude $\vdash \sum_{\{i \in I \,|\, \alpha_i \in B\}} \alpha_i.t_i \sqsupseteq \sum_{\{j \in J \,|\, \beta_j \in B\}} \beta_j.u_j$.

Furthermore, by Lemma 25(3), $\sum_{\{i \in I \,|\, \alpha_i \notin B\}} \alpha_i.t_i \; [+\, \sigma.t_\sigma\,] \gtrsim \mathbf{0} \; [+\, \sigma.u_\sigma\,]$. If $B \neq \emptyset$, one can apply the induction hypothesis to conclude that this relation is also derivable in our axiom system, and we are done. Otherwise, both t and u possess a σ–transition, which yields $\sum_{i \in I} \alpha_i.t_i \; [+\, t_\sigma\,] \gtrsim u_\sigma$ by the definition of $\gtrsim$, with $u_\sigma \equiv \mathbf{0}$ if the summand $\sigma.u_\sigma$ is absent. According to the induction hypothesis (observe that at least one σ is missing when compared to t and u) we obtain $\vdash \sum_{i \in I} \alpha_i.t_i \; [+\, t_\sigma\,] \sqsupseteq u_\sigma$. Hence, we may conclude $\vdash \sum_{i \in I} \alpha_i.t_i \; [+\, \sigma.t_\sigma\,] \sqsupseteq \sigma.(\sum_{i \in I} \alpha_i.t_i) \; [+\, \sigma.t_\sigma\,] \sqsupseteq \sigma.(\sum_{i \in I} \alpha_i.t_i \; [+\, t_\sigma\,]) \sqsupseteq \sigma.u_\sigma \sqsupseteq \mathbf{0} \; [+\, \sigma.u_\sigma\,]$ by Axioms (P5), (P4), and (A4), by the above, and by the fact $\vdash \mathbf{0} + \sigma.\mathbf{0} = \mathbf{0}$ (cf. Axiom (P3) for $P \equiv \mathbf{0}$). □

It is very much desirable to extend our axiomatization to cover parallel composition, too, but this is non–trivial and still an open problem. As already mentioned, $\sigma.a.\mathbf{0} \,|\, \sigma.b.\mathbf{0}$ is strictly faster than $\sigma.a.\sigma.b.\mathbf{0} + \sigma.b.\sigma.a.\mathbf{0}$; but since σ is synchronized, a more sensible expansion law would try to equate $\sigma.a.\mathbf{0} \,|\, \sigma.b.\mathbf{0}$ with $\sigma.(a.\mathbf{0} \,|\, b.\mathbf{0})$. Unfortunately, this law does not hold since the latter process can engage in an a–transition to $\mathbf{0} \,|\, b.\mathbf{0}$ and is therefore strictly faster than the former. Thus, our situation is the same as in Moller and Tofts' paper [15] 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 [15] are all valid in our setting as well; however, we have the additional Axioms (P1) and (P2) which both are valid since σ is just a potential delay that can occur in certain contexts. Also Moller and Tofts do not treat parallel composition completely, just some expansion–like inequalities are listed. Once we know how parallel composition can be dealt with, extending our axiomatization to regular sequential processes, i.e., the class of finite–state sequential processes that do not contain restriction and relabeling operators inside recursion, can be done by adapting Milner's technique for uniquely characterizing recursive processes by systems of equations in normal form [34].

4.3 Abstracting from Internal Computation

The strong faster–than precongruence introduced in Sec. 4.1 is too discriminating for verifying systems in practice. It requires that two systems have to match each other's action transitions exactly, even those labeled with the internal action τ. Consequently, one would like to abstract from τ's and develop a faster–than precongruence from the point of view of an external observer. As our algebra is a derivative of CCS, our approach closely follows the lines of [5].

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 α we define $\hat{\alpha} =_{\text{df}} \epsilon$, if $\alpha = \tau$, and $\hat{\alpha} =_{\text{df}} \alpha$, otherwise. Further, we let $\stackrel{\epsilon}{\Longrightarrow} =_{\text{df}} \stackrel{\tau}{\longrightarrow}^*$ 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 27 (Naive weak faster–than preorder)
A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a naive weak faster–than relation *if the following conditions hold 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 $P \gtrsim_{nv} Q$ 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 $\gtrsim_{nv}$ is not a precongruence. To get closer to our goal to define an observational faster–than precongruence we re–define the third clause of the above definition; please note the analogy to the third clause of Def. 18.

Definition 28 (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) $P \xrightarrow{\alpha} P'$ *implies* $\exists Q'.\, Q \stackrel{\hat{\alpha}}{\Longrightarrow} Q'$ *and* $\langle P', Q'\rangle \in \mathcal{R}$.
(2) $Q \xrightarrow{\alpha} Q'$ *implies* $\exists P'.\, P \stackrel{\hat{\alpha}}{\Longrightarrow} P'$ *and* $\langle P', Q'\rangle \in \mathcal{R}$.
(3) $P \xrightarrow{\sigma} P'$ *implies* $\exists Q', Q'', Q'''.\, Q \stackrel{\epsilon}{\Longrightarrow} Q'' \xrightarrow{\sigma} Q''' \stackrel{\epsilon}{\Longrightarrow} Q', \mathcal{U}(Q'') \subseteq \mathcal{U}(P)$, *and* $\langle P', Q'\rangle \in \mathcal{R}$.

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

While the matching rules for action transitions are the same as in CCS, the one for clock transitions might need some justification due to the inclusion condition on urgent action sets. This condition refers to the processes Q'' and P and *not* to the processes Q and P. The idea is that the clock transition emanating from state Q'' in the weakly matching transition of Q must not be preempted by an urgent communication on an urgent action if it is not preempted in P by such a communication. Intuitively, when matching P, process Q might be able to 'put off' the clock transition finitely often; however, when it does match it, namely in state Q'', it must do so under no greater 'urgent–communications constraint' than P does. From the above definition we may conclude that $\gtrapprox$ is the largest weak faster–than relation and that $\gtrapprox$ is a preorder. In addition, the following proposition holds.

Proposition 29 *The relation* $\gtrapprox$ *is a largest precongruence, for all operators except summation, that is contained in* $\gtrapprox_{nv}$*. (Hence,* $\gtrapprox$ *subsumes the largest precongruence contained in* $\gtrapprox_{nv}$*.)*

The reason for the non–compositionality of the summation operator is similar to that with respect to observational equivalence in CCS [5]. Fortunately, the summation fix used for other bisimulation–based timed process algebras, such as CSA [31], proves effective for TACS, too.

Definition 30 (Weak faster–than precongruence)
A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ *is a* weak faster–than precongruence relation *if the following conditions hold for all* $\langle P, Q\rangle \in \mathcal{R}$ *and* $\alpha \in \mathcal{A}$.

(1) $P \xrightarrow{\alpha} P'$ *implies* $\exists Q'.\, Q \stackrel{\alpha}{\Longrightarrow} Q'$ *and* $P' \gtrapprox Q'$.
(2) $Q \xrightarrow{\alpha} Q'$ *implies* $\exists P'.\, P \stackrel{\alpha}{\Longrightarrow} P'$ *and* $P' \gtrapprox Q'$.
(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 $P \underline{\gtrapprox} Q$ *if* $\langle P, Q\rangle \in \mathcal{R}$ *for some weak faster–than precongruence relation* $\mathcal{R}$.

We first show that $\underline{\gtrapprox}$ is indeed a precongruence and also present a simple full–abstraction result.

Theorem 31 *The relation* $\underset{\simeq}{\sqsupset}$ *is the* largest *precongruence contained in* $\underset{\approx}{\sqsupset}$.

It is also worth pointing out that the strong faster–than precongruence $\underset{\sim}{\sqsupset}$ is included in the weak faster–than precongruence $\underset{\simeq}{\sqsupset}$, which immediately follows by inspecting the respective definitions. With Thm. 31 we are now able to state the main theorem of this section.

Theorem 32 (Full–abstraction)
The relation $\underset{\simeq}{\sqsupset}$ *is the* largest precongruence *contained in* $\underset{\approx}{\sqsupset}_{nv}$.

The validity of this theorem is a consequence of a general result established in universal algebra since (1) $\underset{\approx}{\sqsupset}$ is a preorder contained in $\underset{\approx}{\sqsupset}_{nv}$ (cf. Prop. 29) that comprises the largest precongruence contained in $\underset{\approx}{\sqsupset}_{nv}$, and since (2) $\underset{\simeq}{\sqsupset}$ is the largest precongruence contained in $\underset{\approx}{\sqsupset}$ (cf. Thm. 31).

We leave an axiomatization of our weak faster–than precongruence for future work. It should just be mentioned here that all classical τ–laws are valid for TACS, too, with the exception of the first τ–law for time steps: $\sigma.\tau.P \neq \sigma.P$, where $=$ stands for the kernel of $\underset{\simeq}{\sqsupset}$, since $\tau.P \neq P$.

5 Example

In this section we apply our semantic theory to two examples. The first example is adapted from the one in Moller and Tofts' paper [15] and compares the speeds of different forms of mail delivery. The second example deals with two implementations of a two–place storage. Both examples exercise different features of our theory. While the former applies our axioms for strong faster–than precongruence, the latter relies on the classical bisimulation–style proof–principle for our weak faster–than precongruence.

5.1 Mail Delivery

Consider a fortunate nephew who has three uncles living overseas, all of whom send the nephew a selection of local newspapers *at least* every 14 days. There are two kinds of delivery possible: the quite expensive *air mail* AM which takes at most 2 days to deliver and the cheap *surface mail* SM which might take as long as 10 days. However, for some post–internal reason, occasionally surface–mail items are transported via air mail, too. Moreover, it is important to know that the three uncles come from different stratums: the *rich uncle* RU can always afford the air–mail postage, while the *middle–class uncle* MU only sometimes can and the *poor uncle* PU never can. In our algebra TACS, this situation can be modeled as follows.

$$
\begin{aligned}
\mathrm{RU} &=_{\mathrm{df}} \mu x.(\mathrm{AM} \,|\, \sigma^{14}.x) && \textit{“rich uncle”} \\
\mathrm{MU} &=_{\mathrm{df}} \mu x.((\mathrm{AM} + \mathrm{SM}) \,|\, \sigma^{14}.x) && \textit{“middle–class uncle”} \\
\mathrm{PU} &=_{\mathrm{df}} \mu x.(\mathrm{SM} \,|\, \sigma^{14}.x) && \textit{“poor uncle”}
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{AM} &=_{\mathrm{df}} \mathit{mail}.\sigma^2.\overline{\mathit{deliver}}.\mathbf{0} && \textit{“air mail”} \\
\mathrm{SM} &=_{\mathrm{df}} \mathit{mail}.\sigma^{10}.\overline{\mathit{deliver}}.\mathbf{0} + \mathit{mail}.\sigma^2.\overline{\mathit{deliver}}.\mathbf{0} && \textit{“surface mail”}
\end{aligned}
$$

Intuitively, since we are concerned with worst–case timing behavior, one would expect the process RU to be faster than MU, but MU and PU to be equally fast. Indeed, this turns out to be the case in our setting, as we will show by referring to the axioms of our strong faster–than precongruence.

It is convenient to first establish an auxiliary result, namely that $\mathrm{AM} \underset{\sim}{\sqsupset} \mathrm{SM}$. According to Axiom (P5), $\vdash \sigma^2.\overline{\mathit{deliver}}.\mathbf{0} \sqsupseteq \sigma^{10}.\overline{\mathit{deliver}}.\mathbf{0}$ holds. This implies $\vdash \mathit{mail}.\sigma^2.\overline{\mathit{deliver}}.\mathbf{0} + \mathit{mail}.\sigma^2.\overline{\mathit{deliver}}.\mathbf{0} \sqsupseteq \mathit{mail}.\sigma^{10}.\overline{\mathit{deliver}}.\mathbf{0} + \mathit{mail}.\sigma^2.\overline{\mathit{deliver}}.\mathbf{0}$ by the rules of axiomatic reasoning. Finally, we apply Axiom (A3) in order to obtain $\vdash \mathit{mail}.\sigma^2.\overline{\mathit{deliver}}.\mathbf{0} \sqsupseteq \mathit{mail}.\sigma^{10}.\overline{\mathit{deliver}}.\mathbf{0} + \mathit{mail}.\sigma^2.\overline{\mathit{deliver}}.\mathbf{0}$. Thus, $\vdash \mathrm{AM} \sqsupseteq \mathrm{SM}$ by the definitions of AM and SM. Moreover, using this and Axiom (A3), we have $\vdash \mathrm{AM} = \mathrm{AM} + \mathrm{AM} \sqsupseteq \mathrm{AM} + \mathrm{SM} \sqsupseteq \mathrm{SM} + \mathrm{SM} = \mathrm{SM}$, whence $\mathrm{AM} \underset{\sim}{\sqsupset} \mathrm{AM} + \mathrm{SM} \underset{\sim}{\sqsupset} \mathrm{SM}$ due to the correctness of the axioms (cf. Thm. 26). Because of the compositionality of $\underset{\sim}{\sqsupset}$ with respect to parallel composition and recursion, one may immediately derive $\mathrm{RU} \underset{\sim}{\sqsupset} \mathrm{MU} \underset{\sim}{\sqsupset} \mathrm{PU}$. Since $\underset{\sim}{\sqsupset} \; \subseteq \; \underset{\underline{\sim}}{\sqsupset}$, we thus have $\mathrm{RU} \underset{\underline{\sim}}{\sqsupset} \mathrm{MU} \underset{\underline{\sim}}{\sqsupset} \mathrm{PU}$.

It remains to show that $\mathrm{PU} \underset{\sim}{\sqsupset} \mathrm{MU}$ and that $\mathrm{MU} \not\underset{\sim}{\sqsupset} \mathrm{RU}$. For establishing the former, consider $\vdash \mathrm{SM} = \mathit{mail}.\sigma^{10}.\overline{\mathit{deliver}}.\mathbf{0} + \mathrm{AM} = \mathit{mail}.\sigma^{10}.\overline{\mathit{deliver}}.\mathbf{0} + \mathrm{AM} + \mathrm{AM} = \mathrm{SM} + \mathrm{AM}$ according to the definitions of AM and SM, as well as Axioms (A2) and (A3). This implies $\mathrm{SM} \underset{\sim}{\sqsupset} \mathrm{SM} + \mathrm{AM}$ by Thm. 26 and further $\mathrm{PU} \underset{\sim}{\sqsupset} \mathrm{MU}$ due to the congruence property of $\underset{\sim}{\sqsupset}$. Again, the property $\underset{\sim}{\sqsupset} \; \subseteq \; \underset{\underline{\sim}}{\sqsupset}$ implies $\mathrm{PU} \underset{\underline{\sim}}{\sqsupset} \mathrm{MU}$ as well. For establishing the latter, consider the computation $\mathrm{MU} \xrightarrow{\mathit{mail}} \sigma^{10}.\overline{\mathit{deliver}}.\mathbf{0} \,|\, \sigma^{14}.\mathrm{MU} \xrightarrow{\sigma}{}^2 \sigma^8.\overline{\mathit{deliver}}.\mathbf{0} \,|\, \sigma^{12}.\mathrm{MU} \xrightarrow{\sigma} \sigma^7.\overline{\mathit{deliver}}.\mathbf{0} \,|\, \sigma^{11}.\mathrm{MU}$ of MU, which RU can potentially only match as follows: $\mathrm{RU} \xrightarrow{\mathit{mail}} \sigma^2.\overline{\mathit{deliver}}.\mathbf{0} \,|\, \sigma^{14}.\mathrm{RU} \xrightarrow{\sigma}{}^2 \overline{\mathit{deliver}}.\mathbf{0} \,|\, \sigma^{12}.\mathrm{RU} \xrightarrow{\sigma} \overline{\mathit{deliver}}.\mathbf{0} \,|\, \sigma^{11}.\mathrm{RU}$. However, $\mathcal{U}(\overline{\mathit{deliver}}.\mathbf{0} \,|\, \sigma^{12}.\mathrm{RU}) = \{\overline{\mathit{deliver}}\} \not\subseteq \emptyset = \mathcal{U}(\sigma^8.\overline{\mathit{deliver}}.\mathbf{0} \,|\, \sigma^{12}.\mathrm{MU})$, which shows $\mathrm{MU} \not\underset{\sim}{\sqsupset} \mathrm{RU}$. Analogous reasoning establishes $\mathrm{MU} \not\underset{\underline{\sim}}{\sqsupset} \mathrm{RU}$.

Summarizing, we have (i) $\mathrm{RU} \underset{\underline{\sim}}{\sqsupset} \mathrm{MU} \underset{\underline{\sim}}{\sqsupset} \mathrm{PU}$, (ii) $\mathrm{PU} \underset{\underline{\sim}}{\sqsupset} \mathrm{MU}$, and (iii) $\mathrm{MU} \not\underset{\underline{\sim}}{\sqsupset} \mathrm{RU}$. Hence, process RU is faster than process MU, whereas processes MU and PU are equally fast, as intuitively expected.

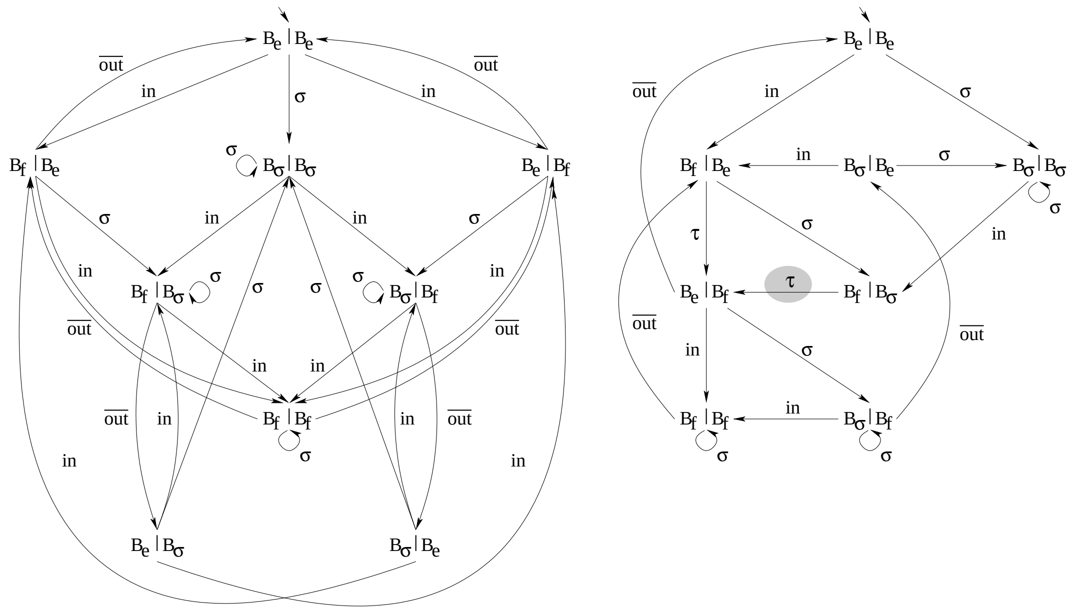

Fig. 1. Semantics of the array variant (left) and the buffer variant (right).

5.2 *Implementation of a 2–place Storage*

This second example deals 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* [5]. 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 (cf. the single clock–prefix in front of action *in*). 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:

$$\text{2ARR} =_{\mathrm{df}} B_e \,|\, B_e \qquad \text{and} \qquad \text{2BUF} =_{\mathrm{df}} (B_e[c/out] \,|\, B_e[c/in]) \setminus \{c\} \;.$$

While 2ARR is simply the (independent) 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 (cf. the auxiliary internal channel c). 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. For the sake of completeness we briefly remark that our buffer formalization does not necessarily preserve the order of items buffered, which is in line with Milner's classical buffer examples for CCS [5].

Table 5
Pairs in the considered weak faster–than relation

$\langle\, (\mathrm{B_e}\vert\mathrm{B_e})\,,\, (\mathrm{B_e}\vert\mathrm{B_e}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_f}\vert\mathrm{B_e})\,,\, (\mathrm{B_f}\vert\mathrm{B_e}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_e}\vert\mathrm{B_f})\,,\, (\mathrm{B_f}\vert\mathrm{B_e}) \setminus \{c\}\rangle$
$\langle\, (\mathrm{B_f}\vert\mathrm{B_e})\,,\, (\mathrm{B_e}\vert\mathrm{B_f}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_f}\vert\mathrm{B_\sigma})\,,\, (\mathrm{B_f}\vert\mathrm{B_\sigma}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_f}\vert\mathrm{B_f})\,,\, (\mathrm{B_f}\vert\mathrm{B_f}) \setminus \{c\}\rangle$
$\langle\, (\mathrm{B_f}\vert\mathrm{B_\sigma})\,,\, (\mathrm{B_\sigma}\vert\mathrm{B_f}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_e}\vert\mathrm{B_\sigma})\,,\, (\mathrm{B_e}\vert\mathrm{B_e}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_\sigma}\vert\mathrm{B_e})\,,\, (\mathrm{B_e}\vert\mathrm{B_e}) \setminus \{c\}\rangle$
$\langle\, (\mathrm{B_\sigma}\vert\mathrm{B_f})\,,\, (\mathrm{B_e}\vert\mathrm{B_f}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_f}\vert\mathrm{B_\sigma})\,,\, (\mathrm{B_e}\vert\mathrm{B_f}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_e}\vert\mathrm{B_f})\,,\, (\mathrm{B_e}\vert\mathrm{B_f}) \setminus \{c\}\rangle$
$\langle\, (\mathrm{B_f}\vert\mathrm{B_\sigma})\,,\, (\mathrm{B_f}\vert\mathrm{B_e}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_\sigma}\vert\mathrm{B_f})\,,\, (\mathrm{B_f}\vert\mathrm{B_e}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_\sigma}\vert\mathrm{B_e})\,,\, (\mathrm{B_\sigma}\vert\mathrm{B_e}) \setminus \{c\}\rangle$
$\langle\, (\mathrm{B_\sigma}\vert\mathrm{B_\sigma})\,,\, (\mathrm{B_\sigma}\vert\mathrm{B_\sigma}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_\sigma}\vert\mathrm{B_f})\,,\, (\mathrm{B_f}\vert\mathrm{B_\sigma}) \setminus \{c\}\rangle$	$\langle\, (\mathrm{B_\sigma}\vert\mathrm{B_f})\,,\, (\mathrm{B_\sigma}\vert\mathrm{B_f}) \setminus \{c\}\rangle$
$\langle\, (\mathrm{B_e}\vert\mathrm{B_\sigma})\,,\, (\mathrm{B_\sigma}\vert\,\mathrm{B_e}) \setminus \{c\}\rangle$		

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_σ 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 τ–transition indicates an urgent internal step of the buffer. Hence, process $(\mathrm{B_f}\vert\mathrm{B_\sigma}) \setminus \{c\}$ cannot engage in a clock transition. The other τ–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. It is easy to check, by employing Def. 28, that this relation is indeed a weak faster–than preorder, whence 2ARR $\underset{\approx}{\sqsupset}$ 2BUF. 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. 30. Thus, 2ARR $\underset{\simeq}{\sqsupset}$ 2BUF, i.e., the 2–place array is faster than the 2–place buffer in all contexts, although functionally equivalent, which matches our abovementioned intuition.

6 Discussion and Related Work

This section highlights the unique features of our approach when compared to related work. There exists a large number of papers on timed process algebras; we refer the reader to [35] for a survey. Usually, these algebras focus on modeling *synchronous* systems, where components are under the regime of a global clock, and do not present faster–than relations. The latter is not surprising because, as argued in [15], it seems unlikely that for synchronous systems a faster–than preorder satisfying a few reasonable properties and being a precongruence for parallel composition exists. Traditionally, timed process algebras aiming at reasoning about synchronous systems have two common features: a delay operator specifying the exact time a process has to wait before it can proceed, and a timeout operator stating which enabled actions are withdrawn and which ones are additionally offered at a particular instant of time. In contrast, our work deals with *asynchronous* systems where actions are not enabled or disabled as time passes. [4] Indeed we added discrete time to CCS simply to evaluate the performance of asynchronous processes and not to increase the functional expressiveness of CCS. We did this by introducing a clock prefix operator specifying a single time bound which we interpreted as upper bound for delays. Some other timed process algebras annotate actions or processes with upper as well as with lower time bounds in the form of timing intervals [6,36]; however, no faster–than relations have been defined in these settings.

Research comparing the worst–case timing behavior of asynchronous systems initially centered around De Nicola and Hennessy's testing theory [22]; it was first conducted within the setting of Petri nets [19,37,21,16] and later for a Theoretical–CSP–style [11] process algebra, called PAFAS [23,24]. 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 analogous testing preorder based on discrete–time semantics [23]. This result depends very much on the testing setting and is different from the sort of discretization obtained for timed automata [17]. 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 σ–prefixes. Also, the equational laws established for the faster–than testing preorder of PAFAS are quite complicated [24], while the simple axioms presented here provide a clear, comprehensive insight into our semantics.

Some researchers consider *testing* [22] to be a more intuitive approach to semantics than *bisimulation* [5]. However, we feel that both are related within our setting. Essentially, the faster–than testing preorder presented for PAFAS in [23] is characterized as inclusion of traces annotated by refusal sets which underly the TACS approach, too. In our faster–than precongruences we require that, when a time step is matched, the urgent action set of the faster process contains the urgent action set of the slower one. One may also say that non–urgent actions can be refused at this moment. If we call a set of non–urgent actions a refusal set, we could replace any clock transition by multiple transitions, one for each refusal set. Then, each refusal–set–transition of the faster process is matched by an equally labeled transition of the slower one.

[4] Of course, while time passes, internal actions —modeling, e.g., local time outs— might be forced to happen, and this can enable and disable actions. For example, process $a.P + \sigma.\tau.Q$ waits at most one time unit before engaging in the τ–action and starting the timeout process Q, i.e., Q is invoked before the second time unit and thus may preempt $a.P$.

Regarding other research concerning faster–than relations, our approach is most closely related to work by Moller and Tofts [15] who developed a bisimulation–based faster–than preorder within the discrete–time process algebra $\ell TCCS$. 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 (cf. Sec. 4.2). It has to be mentioned here that the axioms and the behavioral preorder of Moller and Tofts are not in complete agreement. In fact, writing "σ" for what is actually written "(1)" in [15], $a.\sigma.b.\mathbf{0} + a.b.\mathbf{0}$ is equally fast as $a.b.\mathbf{0}$, which does not seem to be derivable from the axioms; this problem is also acknowledged by Moller [38]. 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 $a.a.b.\mathbf{0}$. The problem seems to lie in the way in which a transition $P \xrightarrow{a} P'$ of a 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 think 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; this should give a better understanding of what worst–case and best–case timing behavior mean for asynchronous systems in (bi)simulation–based settings.

A different idea for relating processes with respect to speed was investigated by Corradini et al. [39] within the so–called ill–timed–but–well–caused approach [40,41]. 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. This way, the timed algebra technically stays very close to untimed ones, but the "ill–timed" runs make the faster–than preorder of Corradini et al. 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 [42,43] or a bisimulation–based setting [44,45]. In all these approaches, except in [42] which does not consider parallel composition, runs of parallel processes are seen to be the interleaved runs of their component processes. Consequently, e.g., process $(\tau.a.\mathbf{0} \,|\, \tau.\overline{a}.b.\mathbf{0}) \setminus \{a\}$ is as efficient as process $\tau.\tau.\tau.b.\mathbf{0}$, whereas in our setting $(\sigma.a.\mathbf{0} \,|\, \sigma.\overline{a}.b.\mathbf{0}) \setminus \{a\}$ is strictly faster than $\sigma.\sigma.\tau.b.\mathbf{0}$.

Finally, it should be mentioned that our approach considers a setting based on discrete time, similar to most related work with the exception of part of Vogler's publications. However, the ideas for the preorders developed here, together with their behavioral theory, can be adapted to continuous time. This is not too difficult but nevertheless requires a substantial amount of work since our language, its semantics, as well as many definitions, lemmas, and theorems need to be suitably adapted or modified. The main insight of this paper, namely the coincidence of the naive, delayed, and indexed faster–than preorders, is thus expected to carry over to the continuous–time setting.

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 several 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 faster–than preorder, including a coarsest precongruence result and an axiomatization for finite sequential processes, and investigated a corresponding "weak" preorder.

Future work should proceed along two orthogonal directions involving both theoretical and practical aspects. From a theory point of view we intend to extend our axiomatization to larger classes of processes and to our weak faster–than precongruence. Recent papers provide an outline how the latter can be done for recursive processes in the presence of preemption [46,47]; as a first step, one could also restrict attention to processes where parallel composition only occurs as top–level operator. Moreover, it remains an open question whether the faster–than precongruence, when defined for continuous time, coincides with the one presented here for discrete time, as is the case in the testing scenario presented in [21]. Currently, we are adapting some of our ideas to comparing the *best–case* efficiency of asynchronous processes, thereby shedding some light onto what worst–case and best–case efficiency means in (bi)simulation–based settings. For putting the novel theory into practice we plan to implement TACS and a decision procedure for our faster–than precongruence in the Concurrency Workbench NC [48], a formal verification tool.

Acknowledgments. We would like to thank the anonymous referees for their excellent comments and suggestions. We also acknowledge Julian Bradfield for suggesting the title of this article.

References

[1] J. Baeten, W. Weijland, Process Algebra, Vol. 18 of Cambridge Tracts in Theoret. Comp. Sc., Cambridge University Press, 1990.

[2] J. Bergstra, A. Ponse, S. Smolka (Eds.), Handbook of Process Algebra, Elsevier Science, 2001.

[3] M. Hennessy, Algebraic Theory of Processes, MIT Press, 1988.

[4] C. Hoare, Communicating Sequential Processes, Prentice Hall, 1985.

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

[6] J. Baeten, J. Bergstra, Real time process algebra, Form. Asp. of Comp. 3 (2) (1991) 142–188.

[7] M. Hennessy, T. Regan, A process algebra for timed systems, Inform. and Comp. 117 (2) (1995) 221–239.

[8] F. Moller, C. Tofts, A temporal calculus of communicating systems, in: CONCUR '90, Vol. 458 of LNCS, Springer-Verlag, 1990, pp. 401–415.

[9] X. Nicollin, J. Sifakis, The algebra of timed processes, ATP: Theory and application, Inform. and Comp. 114 (1994) 131–178.

[10] G. Reed, A. Roscoe, The timed failures-stability model for CSP, TCS 211 (1/2) (1999) 85–127.

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

[12] W. Yi, CCS + time = An interleaving model for real time systems, in: ICALP '91, Vol. 510 of LNCS, Springer-Verlag, 1991, pp. 217–228.

[13] R. Milner, Calculi for synchrony and asynchrony, TCS 25 (3) (1983) 267–310.

[14] G. Berry, G. Gonthier, The ESTEREL synchronous programming language: Design, semantics, implementation, Sc. of Comp. Prog. 19 (2) (1992) 87–152.

[15] F. Moller, C. Tofts, Relating processes with respect to speed, in: CONCUR '91, Vol. 527 of LNCS, Springer-Verlag, 1991, pp. 424–438.

[16] W. Vogler, Efficiency of asynchronous systems, read arcs, and the MUTEX-problem, TCS 275 (1–2) (2002) 589–631.

[17] R. Alur, D. Dill, A theory of timed automata, TCS 126 (2) (1994) 183–235.

[18] N. Lynch, Distributed Algorithms, Morgan Kaufmann Publishers, 1996.

[19] E. Bihler, W. Vogler, Efficiency of token-passing MUTEX-solutions, in: ICATPN '98, Vol. 1420 of LNCS, Springer-Verlag, 1998, pp. 185–204.

[20] L. Jenner, W. Vogler, Fast asynchronous systems in dense time, in: ICALP '96, Vol. 1099 of LNCS, Springer-Verlag, 1996, pp. 75–86.

[21] W. Vogler, Faster asynchronous systems, in: CONCUR '95, Vol. 962 of LNCS, Springer-Verlag, 1995, pp. 299–312.

[22] R. De Nicola, M. Hennessy, Testing equivalences for processes, TCS 34 (1-2) (1984) 83–133.

[23] L. Jenner, W. Vogler, Comparing the efficiency of asynchronous systems, in: ARTS '99, Vol. 1601 of LNCS, Springer-Verlag, 1999, pp. 172–191.

[24] W. Vogler, L. Jenner, Axiomatizing a fragment of PAFAS, ENTCS 39 (3).

[25] F. Corradini, M. Di Berardini, W. Vogler, PAFAS at work: Comparing the worst-case efficiency of three buffer implementations, in: APAQS 2001, CSP, 2001, pp. 231–240.

[26] E. Badouel, P. Darondeau, On guarded recursion, TCS 82 (2) (1991) 403–408.

[27] C. Verhoef, A congruence theorem for structured operational semantics with predicates and negative premises, Nordic J. of Comp. 2 (2) (1995) 274–302.

[28] F. Moller, C. Tofts, Behavioural abstraction in TCCS, in: ICALP '92, Vol. 623 of LNCS, Springer-Verlag, 1992, pp. 559–570.

[29] R. Milner, An algebraic definition of simulation between programs, in: IJCAI '71, William Kaufmann, 1971, pp. 481–489.

[30] D. Park, Concurrency and automata on infinite sequences, in: 5th G.I. Conf. on Theoretical Computer Science, Vol. 104 of LNCS, Springer-Verlag, 1981, pp. 167–183.

[31] R. Cleaveland, G. Lüttgen, M. Mendler, An algebraic theory of multiple clocks, in: CONCUR '97, Vol. 1243 of LNCS, Springer-Verlag, 1997, pp. 166–180.

[32] L. Aceto, W. Fokkink, C. Verhoef, Structural operational semantics, in: Bergstra et al. [2], Ch. 3, pp. 197–292.

[33] L. Aceto, A. Jeffrey, A complete axiomatization of timed bisimulations for a class of timed regular behaviours, TCS 152 (2) (1995) 251–268.

[34] R. Milner, A complete inference system for a class of regular behaviours, J. of Comp. and Sys. Sc. 28 (1984) 439–466.

[35] J. Baeten, C. Middelburg, Process algebra with timing: Real time and discrete time, in: Bergstra et al. [2], Ch. 10, pp. 627–684.

[36] F. Corradini, M. Pistore, Closed interval process algebra versus open interval process algebra, Acta Inform. 37 (7) (2001) 467–509.

[37] L. Jenner, W. Vogler, Fast asynchronous systems in dense time, TCS 254 (1-2) (2001) 379–422.

[38] F. Moller, Private communication (2000).

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

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

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

[42] R. Cleaveland, A. Zwarico, A theory of testing for real time, in: LICS '91, IEEE Computer Society Press, 1991, pp. 110–119.

[43] V. Natarajan, R. Cleaveland, An algebraic theory of process efficiency, in: LICS '96, IEEE Computer Society Press, 1996, pp. 63–72.

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

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

[46] M. Bravetti, R. Gorrieri, A complete axiomatization for observational congruence of prioritized finite-state behaviors, in: ICALP 2000, Vol. 1853 of LNCS, Springer-Verlag, 2000, pp. 744–755.

[47] H. Hermanns, M. Lohrey, Priority and maximal progress are completely axiomatisable, in: CONCUR '98, Vol. 1466 of LNCS, Springer-Verlag, 1998, pp. 237–252.

[48] R. Cleaveland, S. Sims, The NCSU Concurrency Workbench, in: CAV '96, Vol. 1102 of LNCS, Springer-Verlag, 1996, pp. 394–397.

[49] D. Sangiorgi, R. Milner, The problem of 'weak bisimulation up to', in: CONCUR '92, Vol. 630 of LNCS, Springer-Verlag, 1992, pp. 32–46.

A Proofs and Proof Sketches

For completeness, the appendix contains those proofs or proof sketches which were omitted in the main body of the paper in order to enhance the flow of reading.

Proof of Lemma 7. The first two statements can be proved by induction on the inference length of $P' \succ P$. The only interesting case concerns Case (7) of Def. 6, where, for both parts, we can assume $y \neq x$, since x is neither free in $P'[\mu x.P/x]$ nor in $\mu x.P$. Now assume $P'[\mu x.P/x] \succ \mu x.P$ due to $P' \succ P$.

(1) If there exists an unguarded occurrence of y in $\mu x.P$, then there is also one in P and, by induction, in P'. The latter occurrence is also present after substituting $\mu x.P$ for x. Otherwise, y is guarded in $\mu x.P$, in P, and, by induction, in P'. Hence, every free occurrence of y in $P'[\mu x.P/x]$ either stems from P' and is guarded in P', or it is in a subterm of $\mu x.P$, where it is guarded.
(2) By Barendregt's Assumption, we may assume that there is no free occurrence of x in Q and, by induction, $P'[Q/y] \succ P[Q/y]$. As a consequence, we obtain $(P'[\mu x.P/x])[Q/y] \equiv (P'[Q/y])[\mu x.(P[Q/y])/x] \succ \mu x.(P[Q/y]) \equiv (\mu x.P)[Q/y]$.

The other cases are straightforward and, thus, are omitted here.

The proof of the third statement is by induction on the size of Q', including a case analysis on the structure of Q'. The only interesting case is $Q' \equiv \mu y.S$ for some $y \in \mathcal{V}$ and $S \in \widehat{\mathcal{P}}$, where we can assume $P \not\equiv Q$ as well as $y \neq x$, and that y is not free in R. Now, $Q \equiv \mu y.(S[\mu x.R/x])$ and $P \equiv S'[\mu y.S[\mu x.R/x]/y]$ with $S' \succ S[\mu x.R/x]$. By induction hypothesis we can write S' as $S''[\mu x.R/x]$ for some S'' satisfying $S'' \succ S$. We can further write P as $S''[\mu y.S/y][\mu x.R/x]$ since y is not free in R. Finally, we may conclude this case by setting $P' \equiv S''[\mu y.S/y]$. $\square$

Proof sketch of Lemma 13. The proof of this lemma relies on two further lemmas. The first one of these compares the relations $\succ_i$, for all $i \in \mathbb{N}$, to the relation $\succ$; it also compares the relations $\succ_i$ with each other.

Lemma 33

(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* $P, P' \in \widehat{\mathcal{P}}$.

This lemma states some useful facts about our syntactic relations. In particular, Part (3) compares $\succ^{-1}$ with $\succ_i$, for $i > 0$. For validating Part (1) consider $P \succ_i Q$ and show $P \succ_{i+1} Q$ by induction on the inference of $P \succ_i Q$. The proof of Part (2) is analogous; for case $P \succ \sigma.P$ recall that $P \succ P$ and, hence, $P \succ_0 \sigma.P$. Also the proof of Part (3) is analogous; for case $P \succ \sigma.P$ use $P \succ_{i-1} P$ which implies $\sigma.P \succ_i P$. For the latter, the premise $i > 0$ is needed. Finally, observe that Clause 6(7) is matched by Clause 12(7b). $\square$

The second lemma is the analogue of the first two statements of Lemma 7.

Lemma 34 *Let* $P, P', Q \in \widehat{\mathcal{P}}$ *such that* $P' \succ_i P$*, and let* $y \in \mathcal{V}$.

(1) y *is guarded in* P *if and only if* y *is guarded in* P'.
(2) $P'[Q/y] \succ_i P[Q/y]$.

The proof of each statement is similar to the one of the corresponding statement of Lemma 7. In case $P_1 \succ_i \sigma^j.P_n$ (cf. Rule 12(2a)), one needs to use Lemma 7(2) to obtain $P_1[Q/y] \succ \cdots \succ P_n[Q/y]$. $\Box$

We are now able to sketch the proof of Lemma 13. While the proof of Part (1) is obvious, the ones for Parts (2) and (3) are similar to the "functional" part of Prop. 9(2). In Case (2a) we use that $\sigma^j.P_n \xrightarrow{\alpha} P'_n$ if and only if $P_n \xrightarrow{\alpha} P'_n$ if and only if $P_1 \xrightarrow{\alpha} P'_1$ with $P'_1 \succ \cdots \succ P'_n$, where the latter is inferred by Prop. 9(2). In Case (2b) we exploit the property $\succ_i \subseteq \succ_{i+1}$ of Lemma 33(1). The proof for Case (7) is analogous to the one of Prop. 9(2) when using Lemma 34 instead of Lemma 7.

The proof of Part (4) is by induction on the inference length of $Q \succ_0 P$. For Clause (2a) use Lemma 8(2) if $j = 0$. Observe that Clause (2b) does not apply. For Clause (7), employ Lemmas 8(1) and 34(1). $\Box$

Proof of Lemma 14. We first state the following technical property on which our proof relies.

Lemma 35 *If $P_1, P_2, \ldots, P_n \in \widehat{\mathcal{P}}$ for an $n \in \mathbb{N}$ such that $P_1 \succ P_2 \succ \cdots \succ P_n$, and if $P_n \xrightarrow{\sigma} P'$ for some $P' \in \widehat{\mathcal{P}}$, then $P_1 \succ_i P'$, for all $i > 0$.*

The proof is by induction on the structure of P_n. We may assume that all P_i are different and, by Lemma 33(3), that $n > 1$. First observe that P_n cannot be of the form x or $\tau.P$. If P_n is $\mathbf{0}$ or of the form $a.P$, we have $P' \equiv P_n$ and are done by Clause 12(2a) with $j = 0$. If P_n is $\sigma.P$, then $P_1 \succ \cdots \succ P_{n-1} \equiv P \equiv P'$, and we are done by Clauses 12(2a) or (1). The other cases are quite straightforward, except for $P_n \equiv \mu x.Q$. Here, $P_{n-1} \equiv Q'_{n-1}[\mu x.Q/x]$ with $Q'_{n-1} \succ Q$; by Lemma 7(1), x is guarded in Q'_{n-1} since it is guarded in Q. By repeated application of Lemmas 7(1) and (3), we conclude that each P_i, for $1 \leq i \leq n-1$, is of the form $Q'_i[\mu x.Q/x]$ and such that $Q'_1 \succ \cdots \succ Q'_{n-1}$. Furthermore, we have $P' \equiv Q'_n[\mu x.Q/x]$ with $Q \xrightarrow{\sigma} Q'_n$. Now we apply the induction hypothesis to the Q_i's to obtain $Q'_1 \succ_i Q'_n$, which implies $P_1 \equiv Q'_1[\mu x.Q/x] \succ_i Q'_n[\mu x.Q/x] \equiv P'$ by Lemma 34(2). $\Box$

Using the above lemma, we can now formally establish Lemma 14. Both parts are proved by induction on the inference length of $P \succ_i Q$. We only consider the more interesting cases here.

- Part 1:
  (1) For $i > 0$, the time step $P \xrightarrow{\sigma} P'$ implies $P' \succ_j Q \equiv P$, for all j, by Lemmas 33(1) and (2).
  (2a) For $i > 0$, the time step $P_1 \xrightarrow{\sigma} P_0$ implies $P_0 \succ P_1 \succ \cdots \succ P_n$; hence, $P_0 \succ_{i-1} \sigma^j.P_n$. For $i = 0$ and $j > 0$, the same argument shows $P_0 \succ_i \sigma^{j-1}.P_n$, where $\sigma^j.P_n \xrightarrow{\sigma} \sigma^{j-1}.P_n$. For $i = j = 0$, by repeated

application of Prop. 9, $P_1 \xrightarrow{\sigma} P'_1$ implies $P_n \xrightarrow{\sigma} P'_n$ for some P'_n with $P'_1 \succ \cdots \succ P'_n$.

(2b) Observe that $\sigma.P' \xrightarrow{\sigma} P'$ and $P' \succ_i P$ according to the assumption of Def. 12(2b) and that $i+1 > 0$.

The remaining cases are straightforward for $i > 0$. In case of Clause (7) we only have to consider transitions of the form $P'[\mu x.P/x] \xrightarrow{\sigma} P''[\mu x.P/x]$ (by Lemma 2) or $\mu x.P' \xrightarrow{\sigma} P''[\mu x.P'/x]$, where $P'' \succ_{i-1} P$ by induction hypothesis. Then we are done by employing Lemma 34(2) for Clause (7b). Finally, let us consider the case $i = 0$. This is largely analogous using Lemma 34(2) when dealing with Clauses (7a) and (7b). For Clause (3), we apply Lemma 13(4) to deduce that the right–hand side can engage in a time step.

- Part 2:
  (1) $P \xrightarrow{\sigma} P'$ implies $P \succ_{i+1} P'$ by Lemma 33(3).
  (2a) One must use Lemma 35 in case $j = 0$.
  (7) In case of Rule (7a), one employs Lemma 34(2). $\square$

Proof of Theorem 20. The challenging part of this proof is to establish that $\sqsupseteq_{aux'} \subseteq \sqsupseteq_{nv}$. Recall that

$$\sqsupseteq_{aux'} =_{\text{df}} \{\langle P, Q\rangle \mid (P|K_{\mathcal{L}}) \setminus \mathcal{L} \sqsupseteq_{alt,0} (Q|K_{\mathcal{L}}) \setminus \mathcal{L} \quad \text{for some finite } \mathcal{L} \supseteq \mathsf{sort}(P) \cup \mathsf{sort}(Q)\},$$

where

$$K_{\mathcal{L}} =_{\text{df}} \mu x.\, (h.\mathbf{0} + \tau.(\sigma.\tau.\mathbf{0} + f_\sigma.x) + \sum_{a \in \mathcal{L}} \tau.(\tau.\mathbf{0} + f_{\overline{a}}.\mathbf{0} + \overline{a}.x)),$$

and h, f_σ, and $f_{\overline{a}}$, for $a \in \mathcal{L}$, are distinguished actions, i.e., they and their complements are not in $\mathcal{L}$. The idea behind the construction of this context is to turn every visible urgent transition of a process plugged into the context into an urgent τ–transition while not losing its visibility, e.g., the execution of an a–transition is witnessed by the "flag" $f_{\overline{a}}$.

In order to prove the desired statement, it suffices to show that $\sqsupseteq_{aux'}$ is a naive faster–than relation. This can be done similar to the proof of Thm. 19; the most interesting case is when $P \xrightarrow{\sigma} P'$ for some process P' and some $P \sqsupseteq_{aux'} Q$. Hence, $(P|K_{\mathcal{L}}) \setminus \mathcal{L} \sqsupseteq_{alt,0} (Q|K_{\mathcal{L}}) \setminus \mathcal{L}$ for some finite $\mathcal{L} \supseteq \mathsf{sort}(P) \cup \mathsf{sort}(Q)$. We have to establish the existence of some process Q' such that $Q \xrightarrow{\sigma} Q'$ and $P' \sqsupseteq_{aux'} Q'$.

According to the definition of $\sqsupseteq_{alt,0}$, the step $(P|K_{\mathcal{L}}) \setminus \mathcal{L} \xrightarrow{\tau} (P|(\sigma.\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}}))\setminus\mathcal{L}$ of the faster process must be matched by the slower one . Due to the distinguished action f_σ and since $\tau \in \mathcal{U}(K_{\mathcal{L}})$, there are only two possibilities how this can be done:

- Case 1: $(Q|K_{\mathcal{L}}) \setminus \mathcal{L} \xrightarrow{\tau} (Q|(\sigma.\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$ as well as $(P|(\sigma.\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L} \sqsupseteq\!\!\!\!\approx_{\mathit{alt},1} (Q|(\sigma.\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$.

  The faster process $(P|(\sigma.\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$ may now engage in a clock transition to $(P'|(\tau.\mathbf{0}+f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$ which the slower process can only match by a single clock transition to $(Q'|(\tau.\mathbf{0}+f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$, where $Q \xrightarrow{\sigma} Q'$, since this latter process has the urgent internal action τ enabled and can thus not engage in any more time steps.
- Case 2: $(Q|K_{\mathcal{L}}) \setminus \mathcal{L} \xrightarrow{\tau}\xrightarrow{\sigma} (Q'|(\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$, for some $Q' \in \mathcal{P}$ such that $Q \xrightarrow{\sigma} Q'$ and $(P|(\sigma.\tau.\mathbf{0}+f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L} \sqsupseteq\!\!\!\!\approx_{\mathit{alt},1} (Q'|(\tau.\mathbf{0}+f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$. Note that the derived term cannot engage in any further clock transitions due to the urgent internal action.

  At this stage, $(P|(\sigma.\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$ can engage in a clock transition to $(P'|(\tau.\mathbf{0}+f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$. This cannot be matched by $(Q'|(\tau.\mathbf{0}+f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$ via a clock transition, whence $(P|(\sigma.\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$ must have consumed its credit gained in the previous step and $(P'|(\tau.\mathbf{0}+f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L} \sqsupseteq\!\!\!\!\approx_{\mathit{alt},0} (Q'|(\tau.\mathbf{0}+f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$.

As the final step in both cases we consider that $(P'|(\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L}$ can perform its f_σ–transition to $(P'|K_{\mathcal{L}}) \setminus \mathcal{L}$. Since $\tau \in \mathcal{U}((Q'|(\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L})$ and $\tau \in \mathcal{U}(K_{\mathcal{L}})$, this can only be matched by $(Q'|(\tau.\mathbf{0} + f_\sigma.K_{\mathcal{L}})) \setminus \mathcal{L} \xrightarrow{f_\sigma} (Q'|K_{\mathcal{L}}) \setminus \mathcal{L}$ with $(P'|K_{\mathcal{L}}) \setminus \mathcal{L} \sqsupseteq\!\!\!\!\approx_{\mathit{alt},0} (Q'|K_{\mathcal{L}}) \setminus \mathcal{L}$.

In both cases we obtain the existence of some process Q' with $Q \xrightarrow{\sigma} Q'$ and $(P'|K_{\mathcal{L}}) \setminus \mathcal{L} \sqsupseteq\!\!\!\!\approx_{\mathit{alt},0} (Q'|K_{\mathcal{L}}) \setminus \mathcal{L}$. Because of $\mathsf{sort}(P') \subseteq \mathsf{sort}(P)$ and $\mathsf{sort}(Q') \subseteq \mathsf{sort}(Q)$ and thus $\mathcal{L} \supseteq \mathsf{sort}(P') \cup \mathsf{sort}(Q')$, we obtain $P' \sqsupseteq\!\!\!\!\approx_{\mathit{aux}'} Q'$, as desired. □

Proof of Proposition 22. The proof proceeds by induction on the size of process t, i.e., the number of operators contained in t. For the induction base, observe that process $\mathbf{0}$ is trivially in summation form. For the induction step, using Axioms (C1)–(C5) and Axioms (D1)–(D4), one can eliminate restrictions and relabelings as usual [5]. Consequently, t is transformed into a process which is just a sum of prefixed terms. In case of several σ–prefixed terms, these can be merged into one by (repeatedly) applying Axiom (P4) and possibly Axioms (A1) and (A2). Then, the processes trailing the prefixes can be brought into summation form according to the induction hypothesis. The proof details are quite straightforward and, thus, are omitted here. □

Proof of Proposition 24. According to Prop. 22 we may assume t to be in summation form. Now, the proof is by induction on the size of process $t \equiv \sum_{i \in I} \alpha_i.t_i\ [+\ \sigma.t_\sigma\ ]$. In the following, we only comment on the more interesting proof steps and do not explicitly mention applications of Axioms (A1) and (A2). Note that the statement of the proposition is trivially true for the induction base $t \equiv \mathbf{0}$. Moreover, if the optional summand $\sigma.t_\sigma$

does not exist, then one just needs to apply the induction hypothesis to normalize all t_i, for $i \in I$, and the proof is done. Hence, we may assume that the summand $\sigma.t_\sigma$ is present. If Cond. (ii) is violated, i.e., if $\alpha_i = \tau$ for some $i \in I$, then $\vdash t = t' =_{\mathrm{df}} \sum_{i \in I} \alpha_i.t_i + t_\sigma$ by Axiom (P1). Observe that t' is in summation form, has smaller size than t, and satisfies $\mathcal{U}(t) \subseteq \mathcal{U}(t')$. One can now finish off this case by applying the induction hypothesis. Thus, we may assume that Cond. (ii) holds and turn our attention to establishing Cond. (iii). We first (repeatedly) use Axioms (A3) and (P2) and then Axiom (P4) to infer $\vdash \sum_{i \in I} \alpha_i.t_i + \sigma.t_\sigma = \sum_{i \in I} \alpha_i.t_i + \sigma.(\sum_{i \in I} \alpha_i.t_i) + \sigma.t_\sigma = \sum_{i \in I} \alpha_i.t_i + \sigma.(\sum_{i \in I} \alpha_i.t_i + t_\sigma)$. We can now apply the induction hypothesis to process $\sum_{i \in I} \alpha_i.t_i + t_\sigma$ and obtain a term t'' in normal form satisfying $\vdash \sum_{i \in I} \alpha_i.t_i + t_\sigma = t''$ and $\mathcal{U}(\sum_{i \in I} \alpha_i.t_i + t_\sigma) \subseteq \mathcal{U}(t'')$. From this inclusion, it is easy to see that term t'' can be written as $\sum_{k \in K} \gamma_k.t''_k + \sum_{j \in J} \beta_j.t''_j \; [+ \; \sigma.t''_\sigma \;]$, for some index sets K and J, such that $\{\alpha_i \,|\, i \in I\} = \{\gamma_k \,|\, k \in K\}$ and $\{\gamma_k \,|\, k \in K\} \cap \{\beta_j \,|\, j \in J\} = \emptyset$. This implies $(*)$ $\alpha_i \notin \mathcal{I}(\sum_{j \in J} \beta_j.t''_j \; [+ \; \sigma.t''_\sigma])$. By applying the above transformation backwards, i.e., by employing Axioms (P2) and (P4), we infer $\vdash t = \sum_{i \in I} \alpha_i.t_i + \sum_{k \in K} \gamma_k.t''_k + \sigma.(\sum_{j \in J} \beta_j.t''_j \; [+ \; \sigma.t''_\sigma \;])$. The latter term satisfies Cond. (iii) due to property $(*)$ and still satisfies Cond. (ii), too. By induction we can normalize the processes t_i, for $i \in I$, while $\sum_{j \in J} \beta_j.t''_j \; [+ \; \sigma.t''_\sigma \;]$ and the t''_k are in normal form since t'' is. Finally, in case $\sum_{j \in J} \beta_j.t''_j \; [+ \; \sigma.t''_\sigma \;] \equiv \mathbf{0}$, we can eliminate the subterm $\sigma.(\sum_{j \in J} \beta_j.t''_j \; [+ \; \sigma.t''_\sigma \;])$ since $\vdash \mathbf{0} = \mathbf{0} + \sigma.\mathbf{0} = \sigma.\mathbf{0}$ by Axioms (P3) and (A4). This establishes Cond. (i), and we are done. $\square$

Proof of Lemma 25.

- Part (1): If $\alpha_i \equiv \tau$ for some $i \in I$, then the summand $\sigma.t_\sigma$ is not present and the claim follows from Def. 18(2). Otherwise, t can engage in a σ–transition, whence the claim coincides with $\mathcal{U}(u) \subseteq \mathcal{U}(t)$ which follows from Def. 18(3).

We are proving the other two statements separately and proceed along the case distinction suggested by the definition of $\sqsupset\!\!\sim$.

- Part (2): If the right–hand side term can engage in an action transition, say $\sum_{\{j \in J \,|\, \beta_j \in B\}} \beta_j.u_j \xrightarrow{\beta_{j'}} u_{j'}$, then $u \xrightarrow{\beta_{j'}} u_{j'}$ and $t \xrightarrow{\beta_{j'}}$, by the definition of $\sqsupset\!\!\sim$. Since $B \subseteq \{\alpha_i \,|\, i \in I\}$ by (1), we have $\beta_{j'} \equiv \alpha_{i'}$, for some $i' \in I$, such that $t_\sigma \stackrel{\alpha_{i'}}{\not\longrightarrow}$ by Cond. (iii) of normal forms. Hence, $\sum_{\{i \in I \,|\, \alpha_i \in B\}} \alpha_i.t_i \xrightarrow{\alpha_{i'}} t_{i'}$ and $t_{i'} \sqsupset\!\!\sim u_{j'}$. The case where the left–hand side engages in an action transition is analogous. Moreover, it is easy to see that both sides have the same sets of urgent actions and, if τ is not among these actions, then both terms can idle on σ.
- Part (3): The proof of this part is by induction on the size of process u. Since the induction base, i.e., $u \equiv \mathbf{0}$, is trivial, we only focus here on the induction step.

If the left–hand side $\sum_{\{i \in I \,|\, \alpha_i \notin B\}} \alpha_i.t_i \; [+ \; \sigma.t_\sigma \;]$ can engage in an $\alpha_{i'}$–transition to $t_{i'}$, for some $\alpha_{i'} \notin B$, then so can t. Since $\alpha_{i'} \notin B$, the matching $\alpha_{i'}$–transition of u, according to the definition of $\sqsupseteq\!\!\!\!\!\approx$, also exists for the right–hand side $\sum_{\{j \in J \,|\, \beta_j \notin B\}} \beta_j.u_j \; [+ \; \sigma.u_\sigma \;]$. A $\beta_{j'}$–transition of the right–hand side, for $j' \in \{j \in J \,|\, \beta_j \notin B\}$, can be treated analogously.

If the left–hand side can engage in an α–transition to some term t'_σ due to $\sigma.t_\sigma \xrightarrow{\alpha} t'_\sigma$ for some $\alpha \in \mathcal{A}$, then $t \xrightarrow{\alpha} t'_\sigma$ and $\alpha \notin B$ by (1) and Cond. (iii) of normal forms. Hence, the right–hand side can match this transition in the same way as u does according to the definition of $\sqsupseteq\!\!\!\!\!\approx$. A β–transition of the right–hand side, due to $\sigma.u_\sigma \xrightarrow{\beta} u'_\sigma$ for some action β and some term u'_σ, can be dealt with in an analogous fashion.

We now consider $\sum_{\{i \in I \,|\, \alpha_i \notin B\}} \alpha_i.t_i \; [+ \; \sigma.t_\sigma \;] \xrightarrow{\sigma} \sum_{\{i \in I \,|\, \alpha_i \notin B\}} \alpha_i.t_i \; [+ \; t_\sigma \;]$. If $\tau \in B$, then none of the optional summands exists, and $\sum_{\{i \in I \,|\, \alpha_i \notin B\}} \alpha_i.t_i$ and $\sum_{\{j \in J \,|\, \beta_j \notin B\}} \beta_i.u_j$ can idle just as t and u can. If $\tau \notin B$, then $t \xrightarrow{\sigma} \sum_{i \in I} \alpha_i.t_i \; [+ \; t_\sigma \;]$ and, by the definition of $\sqsupseteq\!\!\!\!\!\approx$ and our operational rules: (a) $u \xrightarrow{\sigma} \sum_{j \in J} \beta_j.u_j \,[+ \, u_\sigma \,]$, i.e., $\sum_{\{j \in J \,|\, \beta_j \notin B\}} \beta_j.u_j \,[+ \, \sigma.u_\sigma \,] \xrightarrow{\sigma} \sum_{\{j \in J \,|\, \beta_j \notin B\}} \beta_j.u_j \,[+ \, u_\sigma]$; (b) $\mathcal{U}(u) \subseteq \mathcal{U}(t)$, whence $\mathcal{U}(\sum_{\{j \in J \,|\, \beta_j \notin B\}} \beta_j.u_j \; [+ \; \sigma.u_\sigma \;]) = \mathcal{U}(u) \setminus B \subseteq \mathcal{U}(t) \setminus B = \mathcal{U}(\sum_{\{i \in I \,|\, \alpha_i \notin B\}} \alpha_i.t_i \,[+ \, \sigma.t_\sigma \,])$; (c) $\sum_{i \in I} \alpha_i.t_i \,[+ \, t_\sigma \,] \sqsupseteq\!\!\!\!\!\approx \sum_{j \in J} \beta_j.u_j \; [+ \; u_\sigma \;]$. Since the processes in (c) are in normal form, the induction hypothesis yields $\sum_{\{i \in I \,|\, \alpha_i \notin B\}} \alpha_i.t_i \,[+ \, t_\sigma \,] \sqsupseteq\!\!\!\!\!\approx \sum_{\{j \in J \,|\, \beta_j \notin B\}} \beta_j.u_j \; [+ \; u_\sigma \;]$, as desired. Note that the urgent actions of t_σ and u_σ cannot be in B. $\Box$

Proof of Proposition 29. In the following we prove the precongruence property, i.e., we show that $\sqsupseteq\!\!\!\!\!\approx$ is compositional with respect to action prefixing, clock prefixing, parallel composition, restriction, relabeling, and recursion. Most cases are standard and can be checked along the lines of [5]. The case of clock prefixing is also easy and quite similar to the "strong" case. Therefore, we restrict ourselves to the case of parallel composition. For this proof, the following property turns out to be useful. Let P, P', $Q \in \mathcal{P}$ such that $P \stackrel{\epsilon}{\Longrightarrow} P'$. Then,

$$P|Q \stackrel{\epsilon}{\Longrightarrow} P'|Q \qquad \text{and} \qquad Q|P \stackrel{\epsilon}{\Longrightarrow} Q|P' \tag{A.1}$$

This property can be proved by induction on the "length" of the weak transition $P \stackrel{\epsilon}{\Longrightarrow} P'$. For the compositionality proof regarding parallel composition, it is by Def. 28 sufficient to establish that

$$\mathcal{R} =_{\mathrm{df}} \{\langle P|R, Q|R\rangle \,|\, P \sqsupseteq\!\!\!\!\!\approx Q,\; R \in \mathcal{P}\}$$

is a weak faster–than relation. Let $\langle P|R, Q|R\rangle$ be an arbitrary pair in $\mathcal{R}$.

- *Action transitions:* The cases where $P|R \xrightarrow{\alpha} S$ and $Q|R \xrightarrow{\alpha} S$, for some $S \in \mathcal{P}$ and $\alpha \in \mathcal{A}$ are standard.
- *Clock transitions:* Let $P|R \xrightarrow{\sigma} S$ for some $S \in \mathcal{P}$. By the only applicable Rule (tCom) we know that (i) $P \xrightarrow{\sigma} P'$ for some $P' \in \mathcal{P}$, (ii) $R \xrightarrow{\sigma} R'$ for

some $R' \in \mathcal{P}$, (iii) $\mathcal{U}(P) \cap \overline{\mathcal{U}(R)} = \emptyset$ as well as $\tau \notin \mathcal{U}(P)$ and $\tau \notin \mathcal{U}(R)$, and (iv) $S \equiv P'|R'$. Since $P \gtrapprox Q$, there exist terms $Q', Q'', Q''' \in \mathcal{P}$ such that $Q \stackrel{\epsilon}{\Longrightarrow} Q'' \stackrel{\sigma}{\longrightarrow} Q''' \stackrel{\epsilon}{\Longrightarrow} Q'$, $\mathcal{U}(Q'') \subseteq \mathcal{U}(P)$, and $P' \gtrapprox Q'$. First, observe that $\mathcal{U}(Q'') \cap \overline{\mathcal{U}(R)} \subseteq \mathcal{U}(P) \cap \overline{\mathcal{U}(R)} = \emptyset$ and that $\tau \notin \mathcal{U}(Q'')$. Applying Property (A.1) and Rule (tCom) again, we conclude $Q|R \stackrel{\epsilon}{\Longrightarrow} Q''|R \stackrel{\sigma}{\longrightarrow} Q'''|R' \stackrel{\epsilon}{\Longrightarrow} Q'|R'$. Moreover, $\mathcal{U}(Q''|R) = \mathcal{U}(Q'') \cup \mathcal{U}(R) \subseteq \mathcal{U}(P) \cup \mathcal{U}(R) = \mathcal{U}(P|R)$, since $\tau \notin \mathcal{U}(Q'')$, $\tau \notin \mathcal{U}(P)$, and $\tau \notin \mathcal{U}(R)$. Finally, $\langle P'|R', Q'|R' \rangle \in \mathcal{R}$ holds due to the definition of $\mathcal{R}$.

To conclude this part of the proof, we want to remark that, in order to show $\gtrapprox$ to be compositional with respect to recursion, we need to define a notion of *weak faster–than preorder up to* $\gtrapprox$, which can be done along the lines of [49]:

A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a *weak faster–than relation up to* $\gtrapprox$ 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 $P' \mathcal{R} \gtrapprox Q'$.
(2) $Q \stackrel{\alpha}{\longrightarrow} Q'$ implies $\exists P'.\, P \stackrel{\hat{\alpha}}{\Longrightarrow} P'$ and $P' \gtrapprox \mathcal{R}\, Q'$.
(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 $P' \mathcal{R} \gtrapprox Q'$.

With this definition, the proof is similar to the corresponding proof in the second edition of Milner's book [5].

We are left with establishing that $\gtrapprox$ is a *largest* precongruence, for all operators except summation, that is contained in $\gtrapprox_{nv}$. From universal algebra we know that the *largest* precongruence $\gtrapprox_{nv}^{c'}$ —for all operators except summation— contained in $\gtrapprox_{nv}$ exists. Since $\gtrapprox$ is such a precongruence, the inclusion $\gtrapprox \subseteq \gtrapprox_{nv}^{c'}$ holds. Thus, it remains to show $\gtrapprox_{nv}^{c'} \subseteq \gtrapprox$. Consider the relation

$$\gtrapprox_{aux} =_{\text{df}} \{\langle P, Q \rangle \mid C_{\mathcal{L}}[P] \gtrapprox_{nv} C_{\mathcal{L}}[Q] \text{ for some finite } \mathcal{L} \supseteq \mathsf{sort}(P) \cup \mathsf{sort}(Q)\},$$

where the terms $C_{\mathcal{L}}[x]$ are defined as in the proof of Thm. 19. Since x is simply put in parallel with process $H_{\mathcal{L}}$ in $C_{\mathcal{L}}[x]$, we have that $P \gtrapprox_{nv}^{c'} Q$ implies $C_{\mathcal{L}}[P] \gtrapprox_{nv}^{c'} C_{\mathcal{L}}[Q]$ and $C_{\mathcal{L}}[P] \gtrapprox_{nv} C_{\mathcal{L}}[Q]$; we conclude that $\gtrapprox_{nv}^{c'} \subseteq \gtrapprox_{aux}$. The other necessary inclusion, $\gtrapprox_{aux} \subseteq \gtrapprox$, is established by proving that $\gtrapprox_{aux}$ i a weak faster–than relation. Let $P, Q \in \mathcal{P}$ such that $P \gtrapprox_{aux} Q$, i.e., $C_{\mathcal{L}}[P] \gtrapprox_{nv} C_{\mathcal{L}}[Q]$ for some finite $\mathcal{L} \supseteq \mathsf{sort}(P) \cup \mathsf{sort}(Q)$, and consider the following two situations.

- *Situation 1:* Let $P \stackrel{\alpha}{\longrightarrow} P'$ for some $P' \in \mathcal{P}$ and some $\alpha \in \mathcal{A}$. According to our operational semantics we may derive $C_{\mathcal{L}}[P] \equiv P|H_{\mathcal{L}} \stackrel{\alpha}{\longrightarrow} P'|H_{\mathcal{L}} \equiv C_{\mathcal{L}}[P']$. This transition can only be matched by a corresponding weak transition of Q, say $Q \stackrel{\hat{\alpha}}{\Longrightarrow} Q'$, for some $Q' \in \mathcal{P}$, since only process $H_{\mathcal{L}}$ has the distinguished action e enabled. Therefore, we have $C_{\mathcal{L}}[Q] \equiv Q|H_{\mathcal{L}} \stackrel{\hat{\alpha}}{\Longrightarrow} Q'|H_{\mathcal{L}} \equiv C_{\mathcal{L}}[Q']$ and $C_{\mathcal{L}}[P'] \underset{\approx}{\sqsupset}_{nv} C_{\mathcal{L}}[Q']$. Because $\mathsf{sort}(P') \subseteq \mathsf{sort}(P)$ and $\mathsf{sort}(Q') \subseteq \mathsf{sort}(Q)$, we have $\mathcal{L} \supseteq \mathsf{sort}(P') \cup \mathsf{sort}(Q')$ and thus $P' \underset{\approx}{\sqsupset}_{aux} Q'$. The case where $Q \stackrel{\alpha}{\longrightarrow} Q'$, for some $Q' \in \mathcal{P}$ and some $\alpha \in \mathcal{A}$, is analogous.

- *Situation 2:* Let $P \stackrel{\sigma}{\longrightarrow} P'$ for some $P' \in \mathcal{P}$. As illustrated in the figure below, $C_{\mathcal{L}}[P]$ can perform a τ–transition to $P|H_L$, where $H_L =_{\text{df}} D_L + d_L.H_{\mathcal{L}}$ and $L =_{\text{df}} \{\overline{c} \,|\, c \in (\mathsf{sort}(P) \cup \mathsf{sort}(Q)) \setminus \mathcal{U}(P)\}$; note that $L \subseteq \overline{\mathcal{L}}$. Then, $P|H_L$ can engage in a σ–transition to $P'|H_L$ according to Rule (tCom). Finally, we consider the step $P'|H_L \stackrel{d_L}{\longrightarrow} P'|H_{\mathcal{L}}$.

$$
\begin{array}{ccccc}
P \,| & H_{\mathcal{L}} & \underset{\approx}{\sqsupset}_{nv} & Q \,| & H_{\mathcal{L}} \\
 & \Big\downarrow \tau & & & \Big\Downarrow \epsilon \\
P \,| & (D_L + d_L.H_{\mathcal{L}}) & \underset{\approx}{\sqsupset}_{nv} & Q'' \,| & (D_L + d_L.H_{\mathcal{L}}) \\
\Big\downarrow \sigma & & & \Big\Downarrow \sigma & \\
P' \,| & (D_L + d_L.H_{\mathcal{L}}) & \underset{\approx}{\sqsupset}_{nv} & Q''' \,| & (D_L + d_L.H_{\mathcal{L}}) \\
 & \Big\downarrow d_L & & & \Big\Downarrow d_L \\
P' \,| & H_{\mathcal{L}} & \underset{\approx}{\sqsupset}_{nv} & Q' \,| & H_{\mathcal{L}}
\end{array}
$$

Take a look at the first step. Since $C_{\mathcal{L}}[P] \underset{\approx}{\sqsupset}_{nv} C_{\mathcal{L}}[Q]$, we have $C_{\mathcal{L}}[Q] \stackrel{\epsilon}{\Longrightarrow} W''$, for some $W'' \in \mathcal{P}$. We know that $H_{\mathcal{L}}$ has to perform a τ–transition to H_L but cannot take part in a communication, since e and d_L are distinguished actions. However, Q may be able to perform some τ–transitions to some process $Q'' \in \mathcal{P}$, i.e., $Q \stackrel{\epsilon}{\Longrightarrow} Q''$ and $P|H_L \underset{\approx}{\sqsupset}_{nv} Q''|H_L$.

Now we consider the more interesting second step. Since $P|H_L \underset{\approx}{\sqsupset}_{nv} Q''|H_L$, we know of the existence of some $W''' \in \mathcal{P}$ such that $Q''|H_L \stackrel{\sigma}{\Longrightarrow} W'''$ and $P'|H_L \underset{\approx}{\sqsupset}_{nv} W'''$. According to our operational semantics, Q'' and H_L have to perform a naive temporal weak σ–transition. Since H_L cannot take part in a communication (see above), it can only engage in an idling σ–transition $H_L \stackrel{\sigma}{\longrightarrow} H_L$, and we conclude $W''' \equiv Q'''|H_L$ for some process $Q''' \in \mathcal{P}$ such that $Q'' \stackrel{\sigma}{\Longrightarrow} Q'''$, i.e., $Q'' \stackrel{\epsilon}{\Longrightarrow} Q_1''' \stackrel{\sigma}{\longrightarrow} Q_2''' \stackrel{\epsilon}{\Longrightarrow} Q'''$ for some $Q_1''', Q_2''' \in \mathcal{P}$. Then, $Q''|H_L \stackrel{\epsilon}{\Longrightarrow} Q_1'''|H_L \stackrel{\sigma}{\longrightarrow} Q_2'''|H_L \stackrel{\epsilon}{\Longrightarrow} Q'''|H_L$ must hold. According to Rule (tCom) the condition $\mathcal{U}(Q_1''') \cap \overline{\mathcal{U}(H_L)} = \emptyset$ has to be satisfied in order that the time step may occur. By the choice of L, this condition implies $\mathcal{U}(Q_1''') \subseteq \mathcal{U}(P)$, as desired.

Finally, let $P'|H_L \stackrel{d_L}{\longrightarrow} P'|H_{\mathcal{L}} \equiv C_{\mathcal{L}}[P']$. Since $P'|H_L \mathrel{\underset{\approx}{\sqsupset}}_{nv} Q'''|H_L$, we have $Q'''|H_L \stackrel{d_L}{\Longrightarrow} W'$, for some $W' \in \mathcal{P}$. We know that H_L performs its d_L–transition to $H_{\mathcal{L}}$ since e is a distinguished action. However, Q''' may engage in some τ–transitions to some $Q' \in \mathcal{P}$, i.e., $Q''' \stackrel{\epsilon}{\Longrightarrow} Q'$, and $C_{\mathcal{L}}[P'] \equiv P'|H_{\mathcal{L}} \mathrel{\underset{\approx}{\sqsupset}}_{nv} Q'|H_{\mathcal{L}} \equiv C_{\mathcal{L}}[Q']$.

We have established the existence of processes $Q', Q_1''', Q_2''' \in \mathcal{P}$ such that $Q \stackrel{\epsilon}{\Longrightarrow} Q_1''' \stackrel{\sigma}{\longrightarrow} Q_2''' \stackrel{\epsilon}{\Longrightarrow} Q'$ and $\mathcal{U}(Q_1''') \subseteq \mathcal{U}(P)$. Also $C_{\mathcal{L}}[P'] \mathrel{\underset{\approx}{\sqsupset}}_{nv} C_{\mathcal{L}}[Q']$ holds, as well as $\mathsf{sort}(P') \subseteq \mathsf{sort}(P)$ and $\mathsf{sort}(Q') \subseteq \mathsf{sort}(Q)$, i.e., $P' \mathrel{\underset{\approx}{\sqsupset}}_{aux} Q'$.

Thus, $\mathrel{\underset{\approx}{\sqsupset}}_{aux}$ is indeed a weak faster–than relation, and we are done. □

Proof of Theorem 31. The compositionality of $\mathrel{\underset{\approx}{\sqsupset}}$ is easy to show for the cases of action and clock prefixing, restriction, and relabeling. In the following we deal with the remaining, more interesting cases. Let $P, Q, R, S \in \mathcal{P}$ be such that $P \mathrel{\underset{\approx}{\sqsupset}} Q$ and $R \mathrel{\underset{\approx}{\sqsupset}} S$. Then (1) $P|R \mathrel{\underset{\approx}{\sqsupset}} Q|R$ and (2) $P + R \mathrel{\underset{\approx}{\sqsupset}} Q + R$, which can be established as follows.

(1) According to Def. 30, it is sufficient to prove that the relation

$$\mathcal{R} =_{\mathrm{df}} \{\langle P|R, Q|R\rangle \mid P \mathrel{\underset{\approx}{\sqsupset}} Q\,;\, R \in \mathcal{P}\}$$

is a weak faster–than precongruence relation. Let $\langle P|R, Q|R\rangle \in \mathcal{R}$ be arbitrary.

- *Action transitions:* The cases where $P|R \stackrel{\alpha}{\longrightarrow} S$ or $Q|R \stackrel{\alpha}{\longrightarrow} S$, for some $S \in \mathcal{P}$ and $\alpha \in \mathcal{A}$, are standard.
- *Clock transitions:* Let $P|R \stackrel{\sigma}{\longrightarrow} S$, for some $S \in \mathcal{P}$. This case can easily be treated along the lines of the corresponding case in the proof of the precongruence property of $\mathrel{\underset{\approx}{\sqsupset}}$.

(2) By Def. 30 it is sufficient to establish that the relation

$$\mathcal{R} =_{\mathrm{df}} \{\langle P + R, Q + R\rangle \mid P \mathrel{\underset{\approx}{\sqsupset}} Q\,;\, R \in \mathcal{P}\}$$

is a weak faster–than precongruence relation. Let $\langle P + R, Q + R\rangle \in \mathcal{R}$ be arbitrary.

- *Action transitions:* Let $P + R \stackrel{\alpha}{\longrightarrow} V$, for some $\alpha \in \mathcal{A}$ and $V \in \mathcal{P}$. Since the operational rules for summation with respect to actions are identical to the ones in CCS, and the definition of weak faster–than precongruence coincides with the one of observational congruence in CCS in this particular case, the proof follows along the lines of the corresponding proof in CCS.
- *Clock transitions:* Let $P+R \stackrel{\sigma}{\longrightarrow} V$, for some $V \in \mathcal{P}$, i.e., $P \stackrel{\sigma}{\longrightarrow} P'$ and $R \stackrel{\sigma}{\longrightarrow} R'$ for some $P', R' \in \mathcal{P}$, and $V \equiv P' + R'$ by Rule (tSum). Since $P \mathrel{\underset{\approx}{\sqsupset}} Q$ we know of the existence of some $Q' \in \mathcal{P}$ such that $Q \stackrel{\sigma}{\longrightarrow} Q'$,

$\mathcal{U}(Q) \subseteq \mathcal{U}(P)$, and $P' \gtrapprox Q'$. Therefore, we may conclude $Q+R \stackrel{\sigma}{\longrightarrow} Q'+R'$ by Rule (tSum), as well as $\langle P'+R', Q'+R' \rangle \in \mathcal{R}$ by the definition of $\mathcal{R}$. Moreover, we have $\mathcal{U}(Q+R) = \mathcal{U}(Q) \cup \mathcal{U}(R) \subseteq \mathcal{U}(P) \cup \mathcal{U}(R) = \mathcal{U}(P+R)$ by the definition of urgent action sets, which finishes this part of the proof.

To show that $\gtrapprox$ is compositional with respect to recursion, we have to adapt a notion of "up to" again.

A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a *weak faster–than precongruence relation up to* $\gtrapprox$ if the following conditions hold for every $\langle P, Q \rangle \in \mathcal{R}$ and $\alpha \in \mathcal{A}$.

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

The proof follows pretty much the standard lines (cf. [5]) and, therefore, is omitted here. We are left with establishing the "largest" claim. From universal algebra we know that the *largest* precongruence $\gtrapprox^c$ in $\gtrapprox$ exists and also that $\gtrapprox^c = \{\langle P, Q \rangle \mid \forall C[x].\, C[P] \gtrapprox C[Q]\}$. Since $\gtrapprox$ is a precongruence that is contained in $\gtrapprox$, the inclusion $\gtrapprox \;\subseteq\; \gtrapprox^c$ holds. Thus, it remains to show $\gtrapprox^c \subseteq \gtrapprox$. Consider the relation $\gtrapprox_{aux} =_{\text{df}} \{\langle P, Q \rangle \mid P+c.\mathbf{0} \gtrapprox Q+c.\mathbf{0}\,,$ where $c \notin \mathsf{sort}(P) \cup \mathsf{sort}(Q)\}$. By definition of $\gtrapprox_{aux}$ we have $\gtrapprox^c \subseteq \gtrapprox_{aux}$. We establish the other necessary inclusion $\gtrapprox_{aux} \subseteq \gtrapprox$ by proving that $\gtrapprox_{aux}$ is a weak faster–than precongruence relation. Let $P \gtrapprox_{aux} Q$, i.e., $P+c.\mathbf{0} \gtrapprox Q+c.\mathbf{0}$, and distinguish the following cases.

- *Action transitions:* Let $P \stackrel{\alpha}{\longrightarrow} P'$, i.e., $\alpha \not\equiv c$ and $P+c.\mathbf{0} \stackrel{\alpha}{\longrightarrow} P'$ by Rule (Sum1). Since $P \gtrapprox_{aux} Q$ we conclude the existence of some $V \in \mathcal{P}$ satisfying $Q+c.\mathbf{0} \stackrel{\hat{\alpha}}{\Longrightarrow} V$ and $P' \gtrapprox V$. Because c is a distinguished action we have $V \not\equiv Q$ and, thus, $V \equiv Q'$ and $Q \stackrel{\alpha}{\Longrightarrow} Q'$, for some $Q' \in \mathcal{P}$.
- *Clock transitions:* Let $P \stackrel{\sigma}{\longrightarrow} P'$. By Rules (tAct) and (tSum), $P+c.\mathbf{0} \stackrel{\sigma}{\longrightarrow} P'+c.\mathbf{0}$ holds. Since $P \gtrapprox_{aux} Q$ we know of the existence of some $V, V', V'' \in \mathcal{P}$ such that $Q+c.\mathbf{0} \stackrel{\epsilon}{\Longrightarrow} V' \stackrel{\sigma}{\longrightarrow} V'' \stackrel{\epsilon}{\Longrightarrow} V$, $\mathcal{U}(V') \subseteq \mathcal{U}(P)$, and $P'+c.\mathbf{0} \gtrapprox V$. Because c is a distinguished action not in the sorts of P and Q, we conclude $V' \equiv Q+c.\mathbf{0}$, $V'' \equiv Q'+c.\mathbf{0}$ for some $Q' \in \mathcal{P}$, $V \equiv V''$, $Q \stackrel{\sigma}{\longrightarrow} Q'$, and $\mathcal{U}(Q) \subseteq \mathcal{U}(P)$. Moreover, $P' \gtrapprox_{aux} Q'$ by the definition of $\gtrapprox_{aux}$ and the fact that $\mathsf{sort}(P') \subseteq \mathsf{sort}(P)$ and $\mathsf{sort}(Q') \subseteq \mathsf{sort}(Q)$.

This shows that $\gtrapprox_{aux}$ is a weak faster–than precongruence relation. Hence, $\gtrapprox_{aux} \subseteq \gtrapprox$, as desired. □