Logical Methods in Computer Science
Vol. 9(4:11)2013, pp. 1–34
www.lmcs-online.org

Submitted Jan. 15, 2013
Published Nov. 12, 2013

ALL LINEAR-TIME CONGRUENCES FOR FAMILIAR OPERATORS

ANTTI VALMARI

Department of Mathematics, Tampere University of Technology, Tampere, Finland
e-mail address: Antti.Valmari@tut.fi

Abstract. The detailed behaviour of a system is often represented as a labelled transition system (LTS) and the abstract behaviour as a stuttering-insensitive semantic congruence. Numerous congruences have been presented in the literature. On the other hand, there have not been many results proving the absence of more congruences. This publication fully analyses the linear-time (in a well-defined sense) region with respect to action prefix, hiding, relational renaming, and parallel composition. It contains 40 congruences. They are built from the alphabet, two kinds of traces, two kinds of divergence traces, five kinds of failures, and four kinds of infinite traces. In the case of finite LTSs, infinite traces lose their role and the number of congruences drops to 20. The publication concentrates on the hardest and most novel part of the result, that is, proving the absence of more congruences.

1. Introduction

A sequential program can usually be thought of as computing a partial function from the set of possible inputs to the set of possible outputs. Sometimes the program is not assumed to be deterministic, in which case its meaning is not a partial function but a more general relation. It is widely agreed that relations from inputs to outputs are usually the most appropriate class of mathematical objects for modelling the semantics of sequential programs at the abstract level. Two programs are equivalent if and only if they compute the same relation.

The situation is entirely different with concurrent systems. Process algebra researchers have introduced numerous abstract equivalence notions for comparing the behaviours of systems or subsystems. Many are surveyed in [5]. It is desirable that an equivalence is a congruence, that is, if a subsystem is replaced by an equivalent subsystem, then the system as a whole remains equivalent. Whether or not an equivalence is a congruence depends on the set of operators used in building systems from subsystems. Although the congruence requirement narrows the range down, there is no consensus about which abstract congruence is the most appropriate. Indeed, the abstract congruence that is best for some purpose is not necessarily the best for another purpose.

Behaviours of (sub)systems are often represented as *labelled transition systems*, abbreviated *LTS*. The congruence property makes it possible to apply reductions to subsystems or their LTSs, and thus construct a reduced LTS of the system as a whole that is equivalent

2012 ACM CCS: [**Theory of computation**]: Models of computation—Concurrency.
Key words and phrases: process algebra; semantics; compositionality; verification.

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.2168/LMCS-9(4:11)2013

to the full LTS of the system but often much smaller. This *compositional* approach is a key ingredient in many advanced process-algebraic verification methods, see, e.g., [6, 9, 19].

We say that "$\cong_1$" *implies* "$\cong_2$", if and only if $L \cong_1 L'$ implies $L \cong_2 L'$ for every L and L'. We say that "$\cong_1$" is *weaker* (or coarser) than "$\cong_2$", if and only if "$\cong_2$" implies "$\cong_1$" but not vice versa. We say that "$\cong$" *preserves* a property, if and only if $L \cong L'$ implies that either none or both of L and L' have the property. If, for instance, "$\cong$" preserves deadlocks, L is complicated, L' is simple, and we can reason that $L \cong L'$, then we can analyse the deadlocks of L by analysing the deadlocks of L'. On the other hand, if "$\cong$" also preserves some other information (say, livelocks) about which L and L' disagree, then $L \ncong L'$. In that case, we cannot use L' to analyse the deadlocks of L because we cannot reason that $L \cong L'$. Therefore, we would ideally like to use the weakest possible deadlock-preserving congruence in this analysis task.

Finding the weakest congruence that preserves a given property has been tedious. A handful of such results has been published (e.g., [2–4, 8, 12–14, 17]), but if none of them directly matches, then the user is more or less left with empty hands. Furthermore, to fully exploit the weakest congruence, reduction algorithms have to be adapted to it. The prospect of rewriting the reduction tools for each property is not attractive.

This publication shows that for a significant set of properties and widely accepted set of process operators, the situation is not that bad. This publication simplifies the selection of the abstract congruence that is most appropriate for a task, by listing *all* abstract congruences within a reasonably wide region with respect to a reasonable set of operators. The operators are parallel composition, hiding, relational renaming, and action prefix. The list will make it easy to answer such questions as "what is the weakest congruence that distinguishes $\searrow\!\circ \xrightarrow{a} \circ$ from $\circ \xleftarrow{\tau} \searrow\!\circ \xrightarrow{a} \circ$?"

By *abstract* we mean that invisible actions are not directly observable, although they may have indirect observable consequences. In the vocabulary of linear temporal logic [10], we only consider stuttering-insensitive properties. It is generally accepted that this is a reasonable restriction in the case of concurrent systems. Basically all process-algebraic verification methods make it.

The region that we cover is abstract *linear-time* congruences, in the following sense. A linear-time property holds or fails to hold on an individual complete execution of the system. The system has the property if and only if all its complete executions have it. We originally only consider the execution of visible actions, deadlock, and livelock as directly observable. Then the congruence requirement will bring so-called refusal sets into consideration in the end, but not in the middle, of a sequence of visible actions. The modern version [16] of Hoare's CSP- or failures-divergences equivalence [7] is within our scope, while Milner's observation equivalence or weak bisimilarity [11] is not. Our notion of linear-time is slightly more general than that of the famous stuttering-insensitive linear temporal logic of [10]. This is because we do but the logic does not distinguish deadlock from livelock. The congruence that matches the logic precisely will be found in Section 7. On the other hand, we will see in Section 3 that our notion of linear-time is less general than another line of thought yields.

Two results of this kind were discussed in Chapters 11 and 12 of [16]. With the CSP set of operators and a certain notion of finite linear-time observations, there are only three congruences. Therefore, if the given property meets that notion, to find the weakest congruence that preserves it, it suffices to test the three congruences. If also infinite behaviour

Figure 1: Some simple LTSs. The alphabet of L_C is $\{c_1, c_2\}$. The alphabet of the others is $\Sigma = \{a_1, a_2, \ldots, a_m\}$, where grey notation indicates that, despite the drawing, Σ (and consequently Δ) may be infinite.

is observable, another set of only three congruences is obtained. Our range covers 40 congruences. Four of them are the same as in [16] and two are trivial. The remaining 34 are obtained because we cover a different set of properties and use a smaller set of operators than [16]. The additional two congruences in [16] assume the ability to also observe refusal sets in the middle of a trace.

This publication is based on [20, 21]. The former solved the problem for finite LTSs, finding 20 congruences. The case of infinite LTSs was analysed in [21]. Some of the earlier congruences were split into two and some into three, so the number grew to 40. In [20, 21] and this publication, we concentrate on proving that there are no other congruences than those that we discuss, and skip the proofs that they indeed are congruences.

Section 2 presents the background definitions. Section 3 introduces the strongest abstract linear-time congruence (in our sense). Congruences that are weaker than it are found in Sections 4 to 7. Finally Section 8 summarizes the publication.

2. Basic Definitions

In this publication, systems are composed of labelled transition systems using the action prefix, hiding, relational renaming, and parallel composition operators. In this section we define these and some related concepts, including bisimilarity.

We reserve the symbol τ to denote so-called invisible actions. A *labelled transition system* or *LTS* is the tuple $(S, \Sigma, \Delta, \hat{s})$, where $\tau \notin \Sigma$, $\Delta \subseteq S \times (\Sigma \cup \{\tau\}) \times S$, and $\hat{s} \in S$. We call S the set of *states*, Σ the *alphabet*, Δ the set of *transitions*, and $\hat{s}$ the *initial state*. An LTS is *finite* if and only if its S and Σ (and thus also Δ) are finite. Unless otherwise stated, L_1 denotes the LTS $(S_1, \Sigma_1, \Delta_1, \hat{s}_1)$, and similarly with L, L', L_2, and so on. When we show an LTS as a drawing, *unless otherwise stated, its alphabet is precisely the labels in the drawing excluding* τ. Fig. 1 shows as examples some simple LTSs that are needed later.

LTSs L_1 and L_2 are *bisimilar*, denoted with $L_1 \equiv L_2$, if and only if there is a relation "$\sim$" $\subseteq S_1 \times S_2$ such that

(1) $\Sigma_1 = \Sigma_2$,
(2) $\hat{s}_1 \sim \hat{s}_2$, and
(3) for every $s_1 \in S_1$, $s_2 \in S_2$, $s'_1 \in S_1$, $s'_2 \in S_2$, and $a \in \Sigma \cup \{\tau\}$ such that $s_1 \sim s_2$,
  (a) if $(s_1, a, s'_1) \in \Delta_1$, then there is an s' such that $s'_1 \sim s'$ and $(s_2, a, s') \in \Delta_2$, and
  (b) if $(s_2, a, s'_2) \in \Delta_2$, then there is an s' such that $s' \sim s'_2$ and $(s_1, a, s') \in \Delta_1$.

The relation "$\sim$" is a *bisimulation*.

It is well known that bisimilarity is a very strong equivalence. For the purposes of this publication (and, indeed, almost everywhere in concurrency theory), bisimilar LTSs can be informally thought of as identical. Formal justification for this comes from the fact (whose proof we skip) that replacing an LTS by a bisimilar one in any of our definitions may change the resulting LTS to a bisimilar one but cannot cause any other difference. For instance, if an LTS deadlocks, then also all its bisimilar LTSs deadlock.

Because the purpose of an LTS is to represent the behaviour of a system, it seems intuitively that only the part of the LTS that is reachable from the initial state is significant. Indeed, if L' is the reachable part of L, by letting $s \sim s'$ if and only if $s = s' \in S'$ we see that $L \equiv L'$. So also in our theory, only the reachable part matters.

If Φ is any set of pairs, we define $\mathcal{D}(\Phi) := \{a \mid \exists b : (a,b) \in \Phi\}$ (the *domain*) and $\mathcal{R}(\Phi) := \{b \mid \exists a : (a,b) \in \Phi\}$ (the *range*). We also define $\Phi(a,b) :\Leftrightarrow (a,b) \in \Phi \vee a = b \notin \mathcal{D}(\Phi)$. This definition makes $\Phi(a,a)$ hold whenever a is not in the domain of Φ.

The operators that we use for building systems are defined as follows:

Action prefix: Let $a \neq \tau$. The LTS $L' = a.L$ is defined as $S' = S \cup \{\hat{s}'\}$, where $\hat{s}' \notin S$, $\Sigma' = \Sigma \cup \{a\}$, and $\Delta' = \Delta \cup \{(\hat{s}', a, \hat{s})\}$. That is, $a.L$ executes a and then behaves like L. We do not define $\tau.L$ as we will not need it, but it is clear that it can be built from $a.L$ and the next operator by choosing an a that is not in Σ.

Hiding: Let A be a set. The LTS $L' = L \setminus A$ is defined as $S' = S$, $\Sigma' = \Sigma \setminus A$, $\Delta' = \{(s, a, s') \mid \exists b : (s, b, s') \in \Delta \wedge (a = b \notin A \vee a = \tau \wedge b \in A)\}$, and $\hat{s}' = \hat{s}$. That is, $L \setminus A$ behaves otherwise like L, but all actions in A are replaced by τ.

Relational renaming: Let Φ be a set of pairs such that $\tau \notin \mathcal{D}(\Phi) \cup \mathcal{R}(\Phi)$. The LTS $L' = L\Phi$ is defined as $S' = S$, $\hat{s}' = \hat{s}$, $\Sigma' = \{b \mid \exists a \in \Sigma : \Phi(a,b)\}$, and $\Delta' = \{(s,b,s') \mid \exists a : (s,a,s') \in \Delta \wedge \Phi(a,b)\}$. That is, $L\Phi$ behaves otherwise like L, but the labels of transitions are changed. A label may be replaced by more than one label, resulting in more than one copy of the original transition. If Φ does not specify any new label for a transition, then it keeps its original label. This is in particular the case with τ-transitions.

Parallel composition: The LTS $L = L_1 \mid\mid L_2$ is defined as $S = S_1 \times S_2$, $\Sigma = \Sigma_1 \cup \Sigma_2$, $\hat{s} = (\hat{s}_1, \hat{s}_2)$, and $((s_1, s_2), a, (s_1', s_2')) \in \Delta$ if and only if
(1) $a \notin \Sigma_2$, $(s_1, a, s_1') \in \Delta_1$, and $s_2' = s_2$,
(2) $a \notin \Sigma_1$, $(s_2, a, s_2') \in \Delta_2$, and $s_1' = s_1$, or
(3) $a \in \Sigma_1 \cap \Sigma_2$, $(s_1, a, s_1') \in \Delta_1$, and $(s_2, a, s_2') \in \Delta_2$.
That is, if a belongs to the alphabets of both L_1 and L_2, it is executed simultaneously by both. If $a = \tau$ or a belongs to the alphabet of precisely one of L_1 and L_2, then it is executed by one of L_1 and L_2 while the other stays in the state where it is. Clearly $L_2 \mid\mid L_1 \equiv L_1 \mid\mid L_2$ and $L_1 \mid\mid (L_2 \mid\mid L_3) \equiv (L_1 \mid\mid L_2) \mid\mid L_3$, so we may write $L_1 \mid\mid \cdots \mid\mid L_n$ without confusion.

The CSP language [16] has these operators (and many more), and every major process-algebraic language has at least something similar. Therefore, requiring the congruence property with respect to these operators is justified. One has to keep in mind, however, that if the language does not have all these operators, then it may have more abstract linear-time congruences than the ones in this publication. Indeed, we will see after Theorem 1 that the ability of the renaming operator to convert a single action into many actions is important, and so is the availability of the action prefix operator.

Because the notion of congruence depends on the set of operators and because listing the set in theorems is clumsy, we state the following:

> *In the theorems of this publication, "$\cong$" is a congruence means that it is an equivalence and for all LTSs L and L', if $L \cong L'$, then $a.L \cong a.L'$, $L \setminus A \cong L' \setminus A$, $L\Phi \cong L'\Phi$, $L \mid\mid L'' \cong L' \mid\mid L''$, and $L'' \mid\mid L \cong L'' \mid\mid L'$.*

It follows by structural induction that if $f(L_1,\ldots,L_n)$ is any expression only made of these four operators, and if $L_i \cong L'_i$ for $1 \leq i \leq n$, then $f(L_1,\ldots,L_n) \cong f(L'_1,\ldots,L'_n)$.

3. The Strongest Abstract Linear-time Congruence

In this section, we first define some concepts and notation that are useful for discussing abstract linear-time equivalences. Then we transform the notion of "linear-time" of [10] to the vocabulary of this publication. (Unlike [10], we distinguish between deadlock and livelock.) The resulting abstract equivalence is not a congruence. We analyse what has to be added to make it a congruence. Thanks to the additions, some original information becomes redundant. So we throw it away. We call the result an abstract linear-time congruence, because it does not preserve more information than is necessary to cover linear temporal logic in the sense described above. It is the strongest such congruence, because it does not preserve less information than that. Finally we set the target for the rest of this publication.

For discussing abstract equivalences, it is handy to have notation for talking about paths between states such that only the non-τ labels along the path are shown. Let Σ^* and Σ^ω denote the sets of all finite and infinite sequences of elements of Σ. By $s \xRightarrow{\varepsilon} s'$ we mean that there are $s_0, \ldots, s_n$ such that $s = s_0$, $s_n = s'$, and $(s_{i-1}, \tau, s_i) \in \Delta$ for $1 \leq i \leq n$. By $s \xRightarrow{a_1a_2\cdots a_n} s'$, where $a_1a_2\cdots a_n \in \Sigma^*$, we mean that there are s_0, s'_0, $\ldots$, s_n, s'_n such that $s_0 = s$, $s'_n = s'$, $s_i \xRightarrow{\varepsilon} s'_i$ for $0 \leq i \leq n$, and $(s'_{i-1}, a_i, s_i) \in \Delta$ for $1 \leq i \leq n$. If we do not want to mention s', we write $s \xRightarrow{a_1a_2\cdots a_n}$, and $s \xRightarrow{a_1a_2\cdots}$ denotes the similar notion for infinite sequences $a_1a_2\cdots$. An infinite path can also consist of an uninterrupted infinite sequence of invisible transitions. This is denoted with $s \xrightarrow{\tau^\omega}$.

Let $s \in S$. We say that s is *a deadlock* or *deadlocked* if and only if $\forall a : \forall s' : (s, a, s') \notin \Delta$. We say that s is *stable* if and only if $\forall s' : (s, \tau, s') \notin \Delta$.

An *execution* of L is any path that starts at $\hat{s}$. An execution is *complete* if and only if it is infinite or leads to a deadlock. If an infinite execution only has a finite number of visible actions, then it consists of a finite prefix and a *livelock*, that is, an infinite path only consisting of τ-transitions.

In the linear temporal logic of [10], "linear-time" means that the models of logical formulae are certain kind of abstractions of individual complete executions, and a system satisfies a formula if and only if all its complete executions satisfy it. Analogously, we say that the linear-time semantics of L consists of the complete executions of L. There is, however, one difference: in [10], deadlocking executions are extended to infinite by repeating the last state forever, that is, deadlocks are unified with livelocks. We will not do so, because not unifying them gives a more natural and richer theory, from which the theory with the unification is trivially obtained as a corollary.

The abstract linear-time semantics of L consists of the abstractions of the complete executions of L, that is, *deadlocking traces*, *divergence traces*, and *infinite traces*, defined as follows:

$$\begin{aligned}
D\ell(L) &:= \{\sigma \in \Sigma^* \mid \exists s : \hat{s} \xRightarrow{\sigma} s \wedge \forall a : \forall s' : (s, a, s') \notin \Delta\} \\
Div(L) &:= \{\sigma \in \Sigma^* \mid \exists s : \hat{s} \xRightarrow{\sigma} s \wedge s \xrightarrow{\tau^\omega}\} \\
Inf(L) &:= \{\xi \in \Sigma^\omega \mid \hat{s} \xRightarrow{\xi}\}
\end{aligned}$$

For uniformity, from now on $\Sigma(L)$ denotes the alphabet of L.

We say that *the equivalence induced by* Σ*,* $D\ell$*,* Div*, and* Inf is the one defined by $\Sigma(L) = \Sigma(L') \wedge D\ell(L) = D\ell(L') \wedge Div(L) = Div(L') \wedge Inf(L) = Inf(L')$. Unfortunately, it

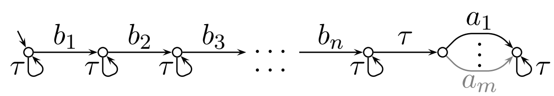


Figure 2: An LTS for detecting the stable failure $(b_1 \cdots b_n, \{a_1, \ldots, a_m\})$.

is not a congruence. To fix this, we define *stable failures*:

$$Sf(L) \;:=\; \{(\sigma, A) \in \Sigma^* \times 2^\Sigma \mid \exists s : \hat{s} =\!\sigma\!\Rightarrow s \wedge \forall a \in A \cup \{\tau\} : \forall s' : (s, a, s') \notin \Delta\}$$

It was proven in [17] that any congruence "$\cong$" that preserves Σ and $D\ell$ also preserves Sf. We repeat the proof here to get familiar with the proof technique. To talk about a finite set $\{a_1, \ldots, a_m\}$ or the infinite set $\{a_1, a_2, \ldots\}$, we use the notation $\{a_1, \ldots, a_m\}$ where "$, a_m$" is grey.

Proof. Assume that $(b_1 \cdots b_n, \{a_1, \ldots, a_m\}) \in Sf(L)$. Let T be the LTS in Fig. 2 with $\Sigma(T) = \Sigma(L)$. By letting L execute $b_1 \cdots b_n$ so that it then refuses $a_1, \ldots, a_m$, and τ, we see that $b_1 \cdots b_n \in D\ell(L \,||\, T)$. Let $L \cong L'$. We have $\Sigma(L') = \Sigma(L)$ because "$\cong$" preserves Σ. By the congruence property $L \,||\, T \cong L' \,||\, T$. That "$\cong$" preserves $D\ell$ yields $b_1 \cdots b_n \in D\ell(L' \,||\, T)$. That is only possible if L' can execute $b_1 \cdots b_n$ such that it then refuses $a_1, \ldots, a_m$, and τ. That is, $(b_1 \cdots b_n, \{a_1, \ldots, a_m\}) \in Sf(L')$. We have proven that $Sf(L) \subseteq Sf(L')$. By symmetry, $Sf(L') \subseteq Sf(L)$. □

Therefore, we must add Sf to the semantics. We have $D\ell(L) = \{\sigma \mid (\sigma, \Sigma) \in Sf(L)\}$. This implies that if $\Sigma(L) = \Sigma(L')$ and $Sf(L) = Sf(L')$, then $D\ell(L) = D\ell(L')$. As a consequence, the equivalence induced by Σ, $D\ell$, Sf, Div, and Inf is the same as the equivalence induced by Σ, Sf, Div, and Inf. That is, we no longer need $D\ell$ as such in the semantics.

The equivalence induced by Σ, Sf, Div, and Inf is a congruence [23]. It is implied by "$\equiv$". It has traditionally been called *chaos-free failures divergences equivalence* or *CFFD-equivalence* for the reason explained in Section 7. We will denote it with "$\doteq$".

Finite, not necessarily complete executions induce *traces*:

$$Tr(L) \;:=\; \{\sigma \in \Sigma^* \mid \hat{s} =\!\sigma\!\Rightarrow\}$$

If $(\sigma, A) \in Sf(L)$, then clearly $(\sigma, \emptyset) \in Sf(L)$ and $\sigma \in Tr(L)$. We will later define also other subsets of $\Sigma^* \times 2^\Sigma$ that have the similar property. With Sf and them, the following notation will be handy:

$$X^{Tr}(L) \;:=\; \{\sigma \mid (\sigma, \emptyset) \in X(L)\}$$

CFFD-equivalence contains full information on traces even without explicitly mentioning them, because of the following easily proven fact:

$$Tr(L) \;=\; Div(L) \cup Sf^{Tr}(L) \tag{3.1}$$

We will also need the following fact.

$$Inf(L) \;\subseteq\; \{a_1 a_2 \cdots \in \Sigma^\omega \mid \forall i : a_1 a_2 \cdots a_i \in Tr(L)\} \tag{3.2}$$

In the case of finite LTSs, even Inf is unnecessary because of the following (see, e.g., [22,23]):

$$Inf(L) \;=\; \{a_1 a_2 \cdots \in \Sigma^\omega \mid \forall i : a_1 a_2 \cdots a_i \in Tr(L)\}\ \text{, if } L \text{ is finite.} \tag{3.3}$$

To summarize, if we define "$\doteq$" as the equivalence induced by Σ, Sf, Div, and Inf, then $L \doteq L'$ also implies $Tr(L) = Tr(L')$ and $D\ell(L) = D\ell(L')$. For finite LTSs, the assumption $Inf(L) = Inf(L')$ is not needed. Because we derived "$\doteq$" by starting with the abstract linear-time semantics and strengthening it only as much as was necessary to make it a congruence, it is reasonable to call it the strongest abstract linear-time congruence.

This is not the only possible use of the phrase "linear-time", however. For instance, one could classify as linear-time everything that can be defined in terms of individual executions and the next-label sets $N(s)$ of the states s along each execution, where $N(s) := \{a \mid \exists s' : (s, a, s') \in \Delta\}$. Now $\mathit{Sf}(L)$ can be rephrased as the set of pairs $(\sigma, A) \in \Sigma^* \times 2^\Sigma$ such that σ leads to a stable state s such that $A \subseteq \Sigma \setminus N(s)$. So "$\doteq$" is linear-time also in this sense. However, so is also the equivalence obtained otherwise similarly, but letting $A = \Sigma \setminus N(s)$. This equivalence is a congruence. It is trivially strictly stronger than "$\doteq$", so it is outside our notion of linear-time.

Our goal is to find all congruences that are implied by "$\doteq$". For any stuttering-insensitive linear-time property in the sense of [10], its optimal congruence is among them.

To break our task into smaller parts, let us consider all possibilities when $\Sigma = \emptyset$. Then $\mathit{Tr}(L) = \{\varepsilon\}$, $\mathit{Inf}(L) = \emptyset$, $\mathit{Sf}(L)$ is either $\emptyset$ or $\{(\varepsilon, \emptyset)\}$, and $\mathit{Div}(L)$ is either $\emptyset$ or $\{\varepsilon\}$. By (3.1) they cannot both be empty. This leaves three possibilities. They can be drawn as follows.

$$\searrow\!\!\circ \qquad\qquad \searrow\!\!\circ\!\circlearrowleft\tau \qquad\qquad \tau\circlearrowright\!\!\searrow\!\!\circ\xrightarrow{\tau}\circ$$

We will study each of the cases $\searrow\!\!\circ \cong \searrow\!\!\circ\!\circlearrowleft\tau$, $\searrow\!\!\circ \cong \tau\circlearrowright\!\!\searrow\!\!\circ\xrightarrow{\tau}\circ \not\cong \searrow\!\!\circ\!\circlearrowleft\tau$, $\searrow\!\!\circ \not\cong \searrow\!\!\circ\!\circlearrowleft\tau \not\cong \tau\circlearrowright\!\!\searrow\!\!\circ\xrightarrow{\tau}\circ \not\cong \searrow\!\!\circ$, and $\searrow\!\!\circ \not\cong \searrow\!\!\circ\!\circlearrowleft\tau \cong \tau\circlearrowright\!\!\searrow\!\!\circ\xrightarrow{\tau}\circ$ in turn.

4. When Deadlock Is Livelock

In this section we find all congruences that are implied by "$\doteq$" and unify deadlock with livelock, that is, have $\searrow\!\!\circ \cong \searrow\!\!\circ\!\circlearrowleft\tau$. Theorems 1, 6, 7, and 4 say that if "$\cong$" preserves any information whatsoever, then it preserves at least Σ; if it preserves more than that, then it also preserves Tr; if it preserves more than that, then it also preserves Inf; and that is all. The technique used in all but one such proofs in this publication is developed and illustrated. It is based on Lemma 3. The section also presents two lemmas related to preserving or not preserving Tr. Theorem 1 is different from others in this section in that it uses a different proof technique and does not make the assumptions mentioned above. So it also applies to bisimulation-based semantics. However, perhaps surprisingly, it depends on the presence of both action prefix and relational renaming in our set of operators.

We define *the dullest congruence* by $L \cong L'$ holds for every L and L'. It is obviously the weakest of all congruences. The next theorem implies that it is the only congruence that does not imply $\Sigma(L) = \Sigma(L')$, that is, preserve Σ. We define $\mathsf{Stop}(A)$ as the 1-state LTS whose alphabet is A and which has no transitions. (So $\mathsf{Stop}(\emptyset) = \searrow\!\!\circ$.)

Theorem 1. If "$\cong$" is implied by "$\equiv$", is a congruence, and does not preserve Σ, then "$\cong$" is the dullest congruence.

Proof. Because "$\cong$" does not preserve Σ, there are LTSs M_1 and M_2 and an a such that $M_1 \cong M_2$ and $a \in \Sigma_1 \setminus \Sigma_2$. Let $C = (\{c\} \cup \Sigma_1 \cup \Sigma_2) \setminus \{a\}$, where $c \neq a$ and $c \neq \tau$. When $i \in \{1, 2\}$, let $f(M_i) = (c.M_i \mid\mid \mathsf{Stop}(\{c\})) \setminus C$. Because $c.M_i$ initially commits to c and $\mathsf{Stop}(\{c\})$ blocks all c-transitions, $f(M_i)$ has no reachable transitions and only one reachable state. C contains all visible actions of $f(M_i)$ except a. So $f(M_1) \equiv \mathsf{Stop}(\{a\})$ and $f(M_2) \equiv \mathsf{Stop}(\emptyset) \equiv \searrow\!\!\circ$. Because $M_1 \cong M_2$, we have $\mathsf{Stop}(\{a\}) \equiv f(M_1) \cong f(M_2) \equiv \searrow\!\!\circ$. This yields $\searrow\!\!\circ \cong \mathsf{Stop}(\{a\})$, because "$\equiv$" implies "$\cong$" and "$\cong$" is an equivalence.

We prove next that each LTS with the empty alphabet is equivalent to $\searrow\!\circ$. Let $L' = (S', \emptyset, \Delta', \hat{s}')$ be an LTS. Let $L'_a = (S', \{a\}, \Delta'_a, \hat{s}')$, where $\Delta'_a = \{(s, a, s') \mid (s, \tau, s') \in \Delta'\}$. By the definition of "$\setminus$", $L' \equiv L'_a \setminus \{a\} \equiv (L'_a \,||\, \searrow\!\circ) \setminus \{a\} \cong (L'_a \,||\, \mathsf{Stop}(\{a\})) \setminus \{a\} \equiv \searrow\!\circ$.

Then we prove that each LTS is equivalent to an LTS with the empty alphabet. Let $L = (S, \Sigma, \Delta, \hat{s})$, $\Phi^{\Sigma}_a = \{a\} \times \Sigma$, $\Delta' = \Delta \cap (S \times \{\tau\} \times S)$, and $L' = (S, \emptyset, \Delta', \hat{s})$. By the definition of "$\Phi$", $\searrow\!\circ \equiv \searrow\!\circ\Phi^{\Sigma}_a \cong \mathsf{Stop}(\{a\})\Phi^{\Sigma}_a \equiv \mathsf{Stop}(\Sigma)$. Therefore, $L \equiv L || \searrow\!\circ \cong L || \mathsf{Stop}(\Sigma) \equiv (S, \Sigma, \Delta', \hat{s}) \equiv L' \,||\, \mathsf{Stop}(\Sigma) \cong L' \,||\, \searrow\!\circ \equiv L'$.

As a conclusion, every LTS is equivalent to $\searrow\!\circ$ and thus to any other. $\square$

This theorem relies on the ability of Φ to convert a single action to an infinite set of actions. Without that ability, the following would be a congruence: $L \cong L'$ if and only if $(\Sigma(L) \setminus \Sigma(L')) \cup (\Sigma(L') \setminus \Sigma(L))$ is finite. Also action prefix is necessary for this theorem. Without it, the following would be a congruence: $L \cong L'$ if and only if $L \equiv L'$ or both $\hat{s} \,{-\tau^\omega\!\rightarrow}$ and $\hat{s}' \,{-\tau^\omega\!\rightarrow}$. That is, initially diverging LTSs could be declared equivalent, even if they had different alphabets.

Theorem 1 says that if a congruence makes any distinctions between LTSs at all, then it preserves at least Σ. On the other hand, it is easy to check from the definitions that the equivalence induced by Σ is a congruence. So it is the second weakest congruence. We have now two congruences that are both trivial.

The next lemma will be needed soon.

Lemma 2. Any congruence that preserves *Inf* also preserves Σ and *Tr*.

Proof. Let "$\cong$" be a congruence that preserves *Inf*. Then $\searrow\!\circ\!\circlearrowleft a \not\cong \searrow\!\circ\!\xrightarrow{a}\!\circ$, so "$\cong$" preserves Σ by Theorem 1. Let $L \cong L'$, $\Sigma = \Sigma(L) = \Sigma(L')$, and $b \notin \Sigma \cup \{\tau\}$. If $\sigma = a_1 a_2 \cdots a_n \in \mathit{Tr}(L)$, then let T^b_σ be $\searrow\!\circ\!\xrightarrow{a_1}\!\circ\!\xrightarrow{a_2} \cdots \xrightarrow{a_n}\!\circ\!\circlearrowleft b$ with the alphabet $\Sigma \cup \{b\}$. We have $\sigma b^\omega \in \mathit{Inf}(L \,||\, T^b_\sigma) = \mathit{Inf}(L' \,||\, T^b_\sigma)$, yielding $\sigma \in \mathit{Tr}(L')$. So $\mathit{Tr}(L) \subseteq \mathit{Tr}(L')$. By symmetry, $\mathit{Tr}(L') \subseteq \mathit{Tr}(L)$. $\square$

The following lemma is central. Many of the subsequent proofs use it. In it, X_1, $\ldots$, X_k are functions from LTSs to sets, like *Tr* and *Sf*.

Lemma 3. Assume that "$\cong$" is an equivalence, is implied by "$\doteq$", and preserves Σ and X_1, $\ldots$, X_k. Assume that there is a function f such that for every LTS L we have $L \cong f(L)$, and $\mathit{Sf}(f(L))$, $\mathit{Div}(f(L))$, and $\mathit{Inf}(f(L))$ can be represented as functions of $\Sigma(L)$ and $X_1(L)$, $\ldots$, $X_k(L)$. Then "$\cong$" is the equivalence induced by Σ and X_1, $\ldots$, X_k.

Proof. Obviously "$\cong$" implies the equivalence induced by Σ and X_1, $\ldots$, X_k.

To prove the implication in the opposite direction, let $\Sigma(L) = \Sigma(L')$ and $X_i(L) = X_i(L')$ for $1 \leq i \leq k$. We need to prove that $L \cong L'$. We have $\Sigma(f(L)) = \Sigma(L) = \Sigma(L') = \Sigma(f(L'))$, because $L \cong f(L)$ and "$\cong$" preserves Σ. When $X \in \{\mathit{Sf}, \mathit{Div}, \mathit{Inf}\}$, let λ_X be the function that represents $X(f(L))$ as was promised. Then $X(f(L)) = \lambda_X(\Sigma(L), X_1(L), \ldots, X_k(L)) = \lambda_X(\Sigma(L'), X_1(L'), \ldots, X_k(L')) = X(f(L'))$. We get $f(L) \doteq f(L')$. So $L \cong f(L) \doteq f(L') \cong L'$ and $L \cong L'$. $\square$

The following proof illustrates, in a simple context, the use of Lemma 3. The f in the proof preserves the congruence and consequently also Σ, *Tr*, and *Inf*. It throws away all information on *Sf* and *Div*, except what can be derived from *Tr* and *Inf* via such facts as $\mathit{Div}(L) \subseteq \mathit{Tr}(L)$. Throwing information away is possible because of the assumption $\searrow\!\circ \cong \searrow\!\circ\!\circlearrowleft \tau$. Although $\mathit{Div}(f(L))$ is neither $\emptyset$ nor $\Sigma(L)^*$, it contains no genuine information, because it is fully determined by $\mathit{Tr}(L)$.

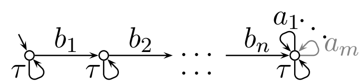


Figure 3: An LTS for detecting the trace $b_1 b_2 \cdots b_n$.

Theorem 4. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves *Inf*, and $\searrow\!\circ \cong \searrow\!\circ\!\circlearrowleft\tau$, then "$\cong$" is the equivalence induced by Σ, *Tr*, and *Inf*.

Proof. By Lemma 2, "$\cong$" preserves Σ and *Tr*. Let $f(L) = L \,||\, \searrow\!\circ\!\circlearrowleft\tau$. We have $L \equiv L \,||\, \searrow\!\circ \cong L \,||\, \searrow\!\circ\!\circlearrowleft\tau = f(L)$. Clearly $\mathit{Sf}(f(L)) = \emptyset$, $\mathit{Div}(f(L)) = \mathit{Tr}(L)$, and $\mathit{Inf}(f(L)) = \mathit{Inf}(L)$. Lemma 3 gives the claim if we choose $k = 2$, $X_1 = \mathit{Tr}$, and $X_2 = \mathit{Inf}$. $\square$

In forthcoming proofs, we will employ renaming and hiding such that precisely those actions synchronize which we want to synchronize. To facilitate that, we introduce the following notation for temporarily attaching an integer i to symbols other than τ. In the notation, $a \neq \tau \notin A$ and $a_j \neq \tau$ for $1 \leq j$.

$$
\begin{array}{rcl}
a^{[i]} & := & (a, i) \\
(a_1 a_2 \cdots a_n)^{[i]} & := & a_1^{[i]} a_2^{[i]} \cdots a_n^{[i]} \\
(a_1 a_2 \cdots)^{[i]} & := & a_1^{[i]} a_2^{[i]} \cdots \\
A^{[i]} & := & \{a^{[i]} \mid a \in A\} \\
\lceil L \rceil^{[i]} & := & L\Phi, \text{ where } \Phi = \{(a, a^{[i]}) \mid a \in \Sigma\} \\
\lfloor L \rfloor_{[i]} & := & L\Phi, \text{ where } \Phi = \{(a^{[i]}, a) \mid a^{[i]} \in \Sigma\}
\end{array}
$$

We will use this notation in the proof of the following lemma, to ensure that certain sets are disjoint.

Lemma 5. If "$\cong$" is a congruence and preserves Σ but not *Tr*, then for any set A such that $\tau \notin A$ there are LTSs M_1^A and M_2^A such that $M_1^A \cong M_2^A$, $\Sigma(M_1^A) = \Sigma(M_2^A) = A$, $\mathit{Sf}(M_1^A) = \mathit{Sf}(M_2^A) = \emptyset$, $\mathit{Tr}(M_1^A) = \mathit{Div}(M_1^A) = A^*$, $\mathit{Tr}(M_2^A) = \mathit{Div}(M_2^A) = \{\varepsilon\}$, $\mathit{Inf}(M_1^A) = A^\omega$, and $\mathit{Inf}(M_2^A) = \emptyset$.

Proof. There are M_1, M_2, and σ such that $M_1 \cong M_2$ and $\sigma \in \mathit{Tr}(M_1) \setminus \mathit{Tr}(M_2)$. Let $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$, $b_1 \cdots b_n = \sigma^{[1]}$, and $\{a_1, \ldots, a_m\} = A^{[2]}$. When $i \in \{1, 2\}$, let

$$M_i^A \;=\; \lfloor\, (T_\sigma \,||\, \lceil M_i \rceil^{[1]}) \setminus \Sigma_M^{[1]} \,\rfloor_{[2]} \;,$$

where $\Sigma(T_\sigma) = \Sigma_M^{[1]} \cup A^{[2]}$ and otherwise T_σ is like in Fig. 3. In the rightmost state of T_σ, there is an a-loop for every $a \in A^{[2]}$. We have $M_1^A \cong M_2^A$ because of the congruence property of "$\cong$". Because $X^{[1]}$ and $Y^{[2]}$ are disjoint for any X and Y, we have $\Sigma(\,(T_\sigma \,||\, \lceil M_i \rceil^{[1]}) \setminus \Sigma_M^{[1]}\,) = (\Sigma_M^{[1]} \cup A^{[2]} \cup \Sigma_M^{[1]}) \setminus \Sigma_M^{[1]} = A^{[2]}$. This yields $\Sigma(M_1^A) = \Sigma(M_2^A) = A$. Because T_σ does not have stable states, we get $\mathit{Sf}(M_1^A) = \mathit{Sf}(M_2^A) = \emptyset$.

Thanks to how renaming and hiding are used, T_σ executes its a-transitions without M_i, while it executes its b-transitions synchronously with M_i and invisibly from the environment. The environment sees the a-transitions with their A-names (instead of $A^{[2]}$-names). Because of the synchronization with T_σ, M_i can only execute τ-transitions and some prefix of σ. Because M_1 can but M_2 cannot execute σ completely, T_σ can reach its rightmost state when in M_1^A but not when in M_2^A. Therefore, $\mathit{Tr}(M_1^A) = \mathit{Div}(M_1^A) = A^*$, $\mathit{Inf}(M_1^A) = A^\omega$, $\mathit{Tr}(M_2^A) = \mathit{Div}(M_2^A) = \{\varepsilon\}$, and $\mathit{Inf}(M_2^A) = \emptyset$. $\square$

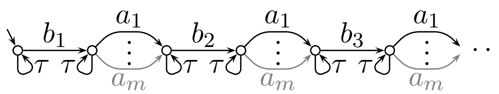


Figure 4: An LTS for detecting the infinite trace $b_1 b_2 \cdots$.

Let $\mathsf{Run}(A)$ denote the LTS whose alphabet is A, which has one state, and whose transitions are $\{(\hat{s}, a, \hat{s}) \mid a \in A\}$ (please see Fig. 1). The following theorem tells that all remaining congruences in this section preserve Σ and Tr.

Theorem 6. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves Σ but not Tr, and $\searrow\!\circ \cong \searrow\!\circ\!\circlearrowleft\tau$, then "$\cong$" is the equivalence induced by Σ.

Proof. Let L be any LTS and $A = \Sigma(L)$. We can reason $\mathsf{Run}(A) \equiv \mathsf{Run}(A) \mid\mid \searrow\!\circ \cong \mathsf{Run}(A) \mid\mid \searrow\!\circ\!\circlearrowleft\tau \doteq M_1^A \cong M_2^A$, where M_1^A and M_2^A are the LTSs in Lemma 5. By choosing $f(L) = L \mid\mid M_2^A$ we get $L \equiv L \mid\mid \mathsf{Run}(A) \cong L \mid\mid M_2^A = f(L)$, so $L \cong f(L)$. Because M_2^A lacks stable failures and blocks all visible actions of L in $L \mid\mid M_2^A$, we have $\mathit{Sf}(f(L)) = \emptyset$, $\mathit{Div}(f(L)) = \{\varepsilon\}$, and $\mathit{Inf}(f(L)) = \emptyset$. They are constants, so Lemma 3 yields the claim if we choose $k = 0$ in it. $\square$

It is widely known that the equivalence induced by Σ and Tr is a congruence. The next theorem says that climbing up the ladder, Inf has to be preserved.

Theorem 7. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves Tr but not Inf, and $\searrow\!\circ \cong \searrow\!\circ\!\circlearrowleft\tau$, then "$\cong$" is the equivalence induced by Σ and Tr.

Proof. There are M_1, M_2, and ξ such that $M_1 \cong M_2$ and $\xi \in \mathit{Inf}(M_1) \setminus \mathit{Inf}(M_2)$. Because "$\cong$" preserves Tr, Theorem 1 implies that it also preserves Σ, so we may let $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$. Let $b_1 b_2 \cdots = \xi^{[1]}$. Let A be any set such that $\tau \notin A$. Let $\{a_1, a_2, \ldots, a_m\} = A^{[2]}$. When $i \in \{1, 2\}$, let

$$M_i^A \;=\; \lfloor (T_\xi \mid\mid \lceil M_i \rceil^{[1]}) \setminus \Sigma_M^{[1]} \rfloor_{[2]} \;,$$

where $\Sigma(T_\xi) = \Sigma_M^{[1]} \cup A^{[2]}$ and otherwise T_ξ is like in Fig. 4. Because $X^{[i]}$ and $Y^{[j]}$ are disjoint whenever $i \neq j$, we have $\Sigma(M_1^A) = \Sigma(M_2^A) = A$. Thanks to the τ-loops in Fig. 4, $\mathit{Sf}(M_1^A) = \mathit{Sf}(M_2^A) = \emptyset$. By (3.2), M_1 can execute any finite prefix of ξ. This yields $\mathit{Tr}(M_1^A) = \mathit{Div}(M_1^A) = A^*$. By the congruence property $M_1^A \cong M_2^A$. Because "$\cong$" preserves Tr, also $\mathit{Tr}(M_2^A) = \mathit{Div}(M_2^A) = A^*$. Since M_1 can but M_2 cannot execute ξ completely, we get $\mathit{Inf}(M_1^A) = A^\omega$ and $\mathit{Inf}(M_2^A) = \emptyset$.

Let L be any LTS and $A = \Sigma(L)$. We can reason $\mathsf{Run}(A) \equiv \mathsf{Run}(A) \mid\mid \searrow\!\circ \cong \mathsf{Run}(A) \mid\mid \searrow\!\circ\!\circlearrowleft\tau \doteq M_1^A \cong M_2^A$, and $L \equiv L \mid\mid \mathsf{Run}(A) \cong L \mid\mid M_2^A$. Lemma 3 gives the claim if we choose $k = 1$, $X_1 = \mathit{Tr}$, and $f(L) = L \mid\mid M_2^A$, because then $L \cong f(L)$, $\mathit{Sf}(f(L)) = \emptyset$, $\mathit{Div}(f(L)) = \mathit{Tr}(L)$, and $\mathit{Inf}(f(L)) = \emptyset$. $\square$

The above proof constructed a function $f(L)$ that throws away all information (modulo "$\doteq$") except Σ and Tr, while preserving "$\cong$". Information on Sf and Div was thrown away using the assumption that $\searrow\!\circ \cong \searrow\!\circ\!\circlearrowleft\tau$. Information on Inf was thrown away by starting with an arbitrary difference on Inf, and amplifying it to a function

$$f'(L, M) \;=\; L \mid\mid \lfloor (T_\xi \mid\mid \lceil M \rceil^{[1]}) \setminus \Sigma_M^{[1]} \rfloor_{[2]}$$

so that $f'(L, M_1)$ preserves $\mathit{Inf}(L)$ while $f'(L, M_2)$ wipes it out. The permission to also throw away all information on Sf and Div simplified the design. We have $L \cong f'(L, M_1) \cong$

$f'(L, M_2) = f(L)$, where the first "$\cong$" takes care of *Sf* and *Div*, and the second of *Inf*. In the construction of f, despite the use of notation defined in this section, ultimately only operators from Section 2 were used.

By Theorem 4, there are no more congruences in this section. In conclusion, altogether precisely four abstract linear-time congruences satisfy $\downarrow\!\circ \cong \downarrow\!\circ\!\circlearrowleft\tau$: those induced by the first zero, one, two, or three of Σ, *Tr*, and *Inf*. That also the last one is a congruence is widely known and proven, e.g., in [23].

5. When Deadlock Is Bothlock Is Not Livelock

In this section we show that only three congruences that are implied by "$\doteq$" satisfy $\downarrow\!\circ \cong \tau\circlearrowright\!\downarrow\!\circ\xrightarrow{\tau}\circ \not\cong \downarrow\!\circ\!\circlearrowleft\tau$. We also introduce an "internal choice" operator $L \sqcap L'$ that will be used in this and later sections. It can be built from the four operators in Section 2, so any equivalence that is a congruence with respect to them also is a congruence with respect to internal choice.

The next theorem tells that all congruences in this section preserve *Sf*. By Theorem 1, they also preserve Σ.

Theorem 8. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and $\tau\circlearrowright\!\downarrow\!\circ\xrightarrow{\tau}\circ \not\cong \downarrow\!\circ\!\circlearrowleft\tau$, then "$\cong$" preserves *Sf*.

Proof. If "$\cong$" does not preserve *Sf*, then there are M_1, M_2, $\sigma = b_1 \cdots b_n$, and $A = \{a_1, \ldots, a_m\}$ such that $M_1 \cong M_2$ and $(\sigma, A) \in \mathit{Sf}(M_1) \setminus \mathit{Sf}(M_2)$. Let $\Sigma_M = \Sigma(M_1)$. If $\Sigma(M_2) \neq \Sigma_M$, then Theorem 1 yields $\downarrow\!\circ\!\circlearrowleft\tau \cong \tau\circlearrowright\!\downarrow\!\circ\xrightarrow{\tau}\circ$. Otherwise, if T^A_σ is the LTS in Fig. 2 with $\Sigma(T^A_\sigma) = \Sigma_M$, we have $(M_2 \,||\, T^A_\sigma) \setminus \Sigma_M \doteq \downarrow\!\circ\!\circlearrowleft\tau$ and $(M_1 \,||\, T^A_\sigma) \setminus \Sigma_M \doteq \tau\circlearrowright\!\downarrow\!\circ\xrightarrow{\tau}\circ$. In both cases, $\downarrow\!\circ\!\circlearrowleft\tau \cong \tau\circlearrowright\!\downarrow\!\circ\xrightarrow{\tau}\circ$, contrary to our assumption. Thus "$\cong$" preserves *Sf*. $\square$

The equivalence induced by Σ and *Sf* is a congruence [23]. However, if the so-called interrupt operator found in CSP or Lotos [1] is employed, then it is no longer a congruence [17].

To prove the next result, the "internal choice" operator of CSP would be handy. It is equivalent to the CCS expression $\tau.P + \tau.Q$. Fortunately, it can be built from our operators.

$$L_1 \sqcap L_2 \quad := \quad \big((\, L_C \,||\, c_1.\lceil L_1 \rceil^{[1]} \,||\, c_2.\lceil L_2 \rceil^{[2]} \,) \setminus \{c_1, c_2\} \big) \Phi \ ,$$

where $c_1 = 1^{[0]}$, $c_2 = 2^{[0]}$, $\Phi = \{(a^{[1]}, a) \mid a \in \Sigma_1\} \cup \{(a^{[2]}, a) \mid a \in \Sigma_2\}$, and L_C has $S_C = \{\hat{s}_C, s_C\}$, $\Sigma_C = \{c_1, c_2\}$, $\Delta_C = \{(\hat{s}_C, c_1, s_C), (\hat{s}_C, c_2, s_C)\}$, and $\hat{s}_C \neq s_C$ (please see Fig. 1). (Here c_1 and c_2 could be any distinct symbols that are not in $\Sigma_1^{[1]} \cup \Sigma_2^{[2]}$.)

The CFFD-semantics of this operator is simple:

$$\begin{aligned}
\Sigma(L \sqcap L') &= \Sigma(L) \cup \Sigma(L') \\
\mathit{Sf}(L \sqcap L') &= \mathit{Sf}(L) \cup \mathit{Sf}(L') \\
\mathit{Div}(L \sqcap L') &= \mathit{Div}(L) \cup \mathit{Div}(L') \\
\mathit{Inf}(L \sqcap L') &= \mathit{Inf}(L) \cup \mathit{Inf}(L')
\end{aligned}$$

The next congruence in this section also preserves *Tr*.

Theorem 9. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves *Sf* but not *Tr*, and $\downarrow\!\circ \cong \tau\circlearrowright\!\downarrow\!\circ\xrightarrow{\tau}\circ$, then "$\cong$" is the equivalence induced by Σ and *Sf*.

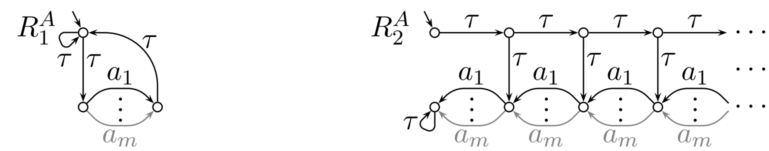


Figure 5: R_1^A has $\Sigma(R_1^A) = A = \{a_1, \ldots, a_m\}$, $\mathit{Sf}(R_1^A) = A^* \times \{\emptyset\}$, $\mathit{Div}(R_1^A) = A^*$, and $\mathit{Inf}(R_1^A) = A^\omega$. R_2^A has the same except $\mathit{Inf}(R_2^A) = \emptyset$.

Proof. Let L be any LTS and $A = \Sigma(L)$. By Theorem 1, "$\cong$" preserves Σ. The assumptions of Lemma 5 hold, so we can use its M_1^A and M_2^A. Let $f(L) = (L \,||\, \tau\circlearrowleft\!\!\downarrow\!\circ \xrightarrow{\tau} \circ) \sqcap M_1^A$. We have $L \equiv L \,||\, \searrow\!\circ \cong L \,||\, \tau\circlearrowleft\!\!\downarrow\!\circ \xrightarrow{\tau} \circ \doteq (L \,||\, \tau\circlearrowleft\!\!\downarrow\!\circ \xrightarrow{\tau} \circ) \sqcap M_2^A$, so $L \cong f(L)$. Furthermore, $\mathit{Sf}(f(L)) = \mathit{Sf}(L)$, $\mathit{Div}(f(L)) = A^* = \Sigma(L)^*$, and $\mathit{Inf}(f(L)) = A^\omega = \Sigma(L)^\omega$. With $k = 1$ and $X_1 = \mathit{Sf}$, Lemma 3 gives the claim. □

The equivalence induced by Σ, Tr, and Sf is a congruence [23].

At the next level, also Inf has to be preserved. To prove this, we need a more complicated construction than in the proof of Theorem 7, because this time Sf has to be preserved.

Theorem 10. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves Tr and Sf but not Inf, and $\searrow\!\circ \cong \tau\circlearrowleft\!\!\downarrow\!\circ \xrightarrow{\tau} \circ$, then "$\cong$" is the equivalence induced by Σ, Tr, and Sf.

Proof. Let $M_1 \cong M_2$, $\xi \in \mathit{Inf}(M_1) \setminus \mathit{Inf}(M_2)$, $b_1 b_2 \cdots = \xi^{[1]}$, and A be any set such that $\tau \notin A$. By Theorem 1, "$\cong$" preserves Σ. Let $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$. When $i \in \{1, 2\}$, let

$$M_i^A \;=\; \lfloor (T_\xi \,||\, \lceil M_i \rceil^{[1]}) \setminus \Sigma_M^{[1]} \rfloor_{[2]} \;,$$

where $\Sigma(T_\xi) = \Sigma_M^{[1]} \cup A^{[2]}$ and otherwise T_ξ is like in Fig. 4.

Because T_ξ does not have stable states, we have $\mathit{Sf}(M_1^A) = \mathit{Sf}(M_2^A) = \emptyset$. Because $\lceil M_2 \rceil^{[1]}$ lacks the infinite trace $b_1 b_2 \cdots$, we have $\mathit{Inf}(M_2^A) = \emptyset$. Let R_1^A and R_2^A be the LTSs in Fig. 5. We have $\mathit{Div}(R_2^A) = A^*$. These imply $M_2^A \sqcap R_2^A \doteq R_2^A$. On the other hand, $\mathit{Inf}(M_1^A) = \mathit{Inf}(R_1^A) = A^\omega$, $\mathit{Sf}(R_1^A) = \mathit{Sf}(R_2^A)$, and also $\mathit{Div}(R_1^A) = A^*$, so $M_1^A \sqcap R_2^A \doteq R_1^A$. As a consequence, $R_1^A \doteq M_1^A \sqcap R_2^A \cong M_2^A \sqcap R_2^A \doteq R_2^A$.

By choosing $A = \Sigma(L)$ and $f(L) = L \,||\, R_2^A$ we get $L \equiv L \,||\, \searrow\!\circ \cong L \,||\, \tau\circlearrowleft\!\!\downarrow\!\circ \xrightarrow{\tau} \circ \doteq L \,||\, R_1^A \cong L \,||\, R_2^A$, so $L \cong f(L)$. We have $\mathit{Sf}(f(L)) = \mathit{Sf}(L)$, $\mathit{Div}(f(L)) = \mathit{Tr}(f(L)) = \mathit{Tr}(L)$, and $\mathit{Inf}(f(L)) = \emptyset$. With $k = 2$, $X_1 = \mathit{Tr}$, and $X_2 = \mathit{Sf}$, Lemma 3 gives the claim. □

The equivalence induced by Σ, Tr, Sf, and Inf is the intersection of the equivalences induced by $(\Sigma, \mathit{Tr}, \mathit{Sf})$ and $(\Sigma, \mathit{Tr}, \mathit{Inf})$. So it is the intersection of two congruences and thus a congruence. We now show that it is the last one in this section.

Theorem 11. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves Sf and Inf, and $\searrow\!\circ \cong \tau\circlearrowleft\!\!\downarrow\!\circ \xrightarrow{\tau} \circ$, then "$\cong$" is the equivalence induced by Σ, Tr, Sf, and Inf.

Proof. By Lemma 2, "$\cong$" preserves Σ and Tr. Let $f(L) = L \,||\, \tau\circlearrowleft\!\!\downarrow\!\circ \xrightarrow{\tau} \circ$. We have $L \equiv L \,||\, \searrow\!\circ \cong L \,||\, \tau\circlearrowleft\!\!\downarrow\!\circ \xrightarrow{\tau} \circ = f(L)$. Clearly $\mathit{Sf}(f(L)) = \mathit{Sf}(L)$, $\mathit{Div}(f(L)) = \mathit{Tr}(L)$, and $\mathit{Inf}(f(L)) = \mathit{Inf}(L)$. Letting $k = 3$, $X_1 = \mathit{Tr}$, $X_2 = \mathit{Sf}$, and $X_3 = \mathit{Inf}$, Lemma 3 gives the claim. □

To summarize, precisely three abstract linear-time congruences satisfy $\searrow\!\circ \cong \tau\circlearrowleft\!\!\downarrow\!\circ \xrightarrow{\tau} \circ \not\cong \searrow\!\circ\!\circlearrowleft\tau$: those induced by (Σ, Sf), $(\Sigma, \mathit{Tr}, \mathit{Sf})$, and $(\Sigma, \mathit{Tr}, \mathit{Sf}, \mathit{Inf})$.

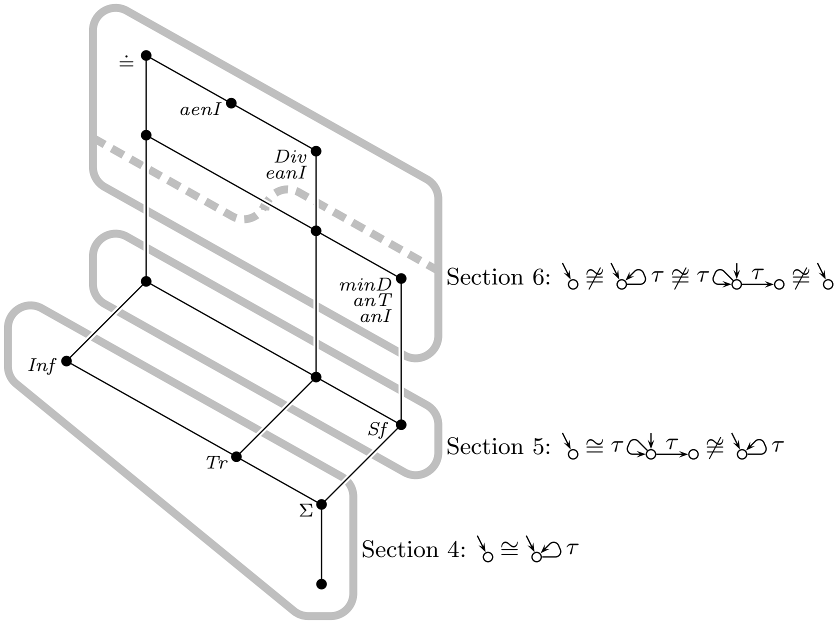


Figure 6: The congruences in Sections 4, 5, and 6 as a Hasse diagram. Names in *italics* indicate the new preserved set(s).

6. When All Three Are Non-equivalent

Figure 6 shows the results of the previous two sections and this section. In this section we survey the region where $\not\cong$ τ $\not\cong$ τ τ $\not\cong$. We need new semantic sets. They are defined in Subsection 6.1. Many proofs in this and the next section treat end states of divergence traces differently from end states of other traces. For this to be possible, no state must be simultaneously the end state of both a divergence trace and a nondivergent trace. Subsection 6.3 presents a construct with which LTSs can be transformed into such a form, while preserving bisimilarity. The theorems that there are no other congruences are presented in Subsections 6.2 and 6.4.

6.1. **New kinds of divergence and infinite traces.** In this subsection we define new semantic sets that are related to divergence traces or infinite traces, and briefly study their properties.

Minimal divergence traces minD are divergence traces whose proper prefixes are not divergence traces. Finite extensions of minimal divergence traces *extT* are an alternative representation for the same information (assuming that Σ is available). Also infinite extensions *extI* can be derived from *minD*. *Always-nondivergent traces anT* are traces which and whose proper prefixes are not divergence traces, and similarly with *always-nondivergent infinite traces anI*. *Eventually-always-nondivergent infinite traces eanI* may have a finite number of divergence traces as prefixes. *Always-eventually-nondivergent infinite traces aenI* have an infinite number of prefixes that are not divergence traces.

$$\begin{array}{rcl}
minD(L) & := & \{a_1 \cdots a_n \in Div(L) \mid \forall i; 0 \le i < n : a_1 \cdots a_i \notin Div(L)\} \\
extT(L) & := & \{a_1 \cdots a_n \in \Sigma(L)^* \mid \exists i; 0 \le i \le n : a_1 \cdots a_i \in minD(L)\} \\
extI(L) & := & \{a_1 a_2 \cdots \in \Sigma(L)^\omega \mid \exists i; i \ge 0 : a_1 \cdots a_i \in minD(L)\} \\
anT(L) & := & Tr(L) \setminus extT(L) \\
anI(L) & := & Inf(L) \setminus extI(L) \\
eanI(L) & := & \{a_1 a_2 \cdots \in Inf(L) \mid \exists n; n \ge 0 : \forall i; i \ge n : a_1 \cdots a_i \notin Div(L)\} \\
aenI(L) & := & \{a_1 a_2 \cdots \in Inf(L) \mid \forall n; n \ge 0 : \exists i; i \ge n : a_1 \cdots a_i \notin Div(L)\}
\end{array}$$

We have

$$\begin{array}{rcll}
minD(L) & = & \{a_1 \cdots a_n \in extT(L) \mid n = 0 \vee a_1 \cdots a_{n-1} \notin extT(L)\} & , \\
anT(L) & = & Sf^{Tr}(L) \setminus extT(L) & , \text{and} \\
anI(L) & \subseteq & eanI(L) \ \subseteq \ aenI(L) \ \subseteq \ Inf(L) & .
\end{array}$$

Lemma 12. Any congruence that preserves $minD$ also preserves Σ and anT.

Proof. By Theorem 1, it preserves Σ. Let $L \cong L'$ and $b \notin \Sigma(L) \cup \{\tau\}$. For each $\sigma = a_1 \cdots a_n \in \Sigma(L)^*$, let T_σ be the LTS whose graph is $\searrow\!\circ \xrightarrow{a_1} \circ \xrightarrow{a_2} \ldots \xrightarrow{a_n} \circ \xrightarrow{b} \circ \circlearrowleft \tau$ and whose alphabet is $\Sigma(L) \cup \{b\}$. We have $\sigma \in Tr(L)$ if and only if $\sigma b \in Div(L \,||\, T_\sigma)$. If $0 \le i \le n$, then $a_1 \cdots a_i \in Div(L)$ if and only if $a_1 \cdots a_i \in Div(L \,||\, T_\sigma)$. Therefore, $\sigma \in anT(L)$ if and only if $\sigma b \in minD(L \,||\, T_\sigma)$ if and only if $\sigma b \in minD(L' \,||\, T_\sigma)$ if and only if $\sigma \in anT(L')$. $\square$

Lemma 13. Any congruence that preserves $minD$ also preserves Σ and anI.

Proof. By Theorem 1, it preserves Σ. Let $L \cong L'$, $a_1 a_2 \cdots \in anI(L)$, and $T = \searrow\!\circ \xrightarrow{a_1} \circ \xrightarrow{a_2} \ldots$ with $\Sigma(T) = \Sigma(L)$. None of $a_1 \cdots a_i$ is in $minD(L) = minD(L')$, yielding $minD(L' \,||\, T) = \emptyset$. On the other hand, $\varepsilon \in minD(\,(L \,||\, T) \setminus \Sigma(L)\,) = minD(\,(L' \,||\, T) \setminus \Sigma(L')\,)$. So $a_1 \cdots a_i \notin Div(L' \,||\, T)$, $a_1 a_2 \cdots \in Inf(L' \,||\, T)$, $a_1 a_2 \cdots \in Inf(L')$, and $a_1 a_2 \cdots \in anI(L')$. $\square$

Lemma 14. Any congruence that preserves Div also preserves Tr.

Proof. $\sigma \in Tr(L) \Leftrightarrow \sigma \in Div(L \,||\, \searrow\!\circ \circlearrowleft \tau)$. $\square$

Lemma 15. Any congruence that preserves Div also preserves Σ and $eanI$.

Proof. By Theorem 1, it preserves Σ. Let $L \cong L'$ and $\xi \in eanI(L)$. If no prefix of ξ is in $Div(L)$, then let $i = 1$, and otherwise let i be 2 plus the length of the longest prefix of ξ that is in $Div(L)$. Let $a_i \notin \Sigma(L) \cup \{\tau\}$ and, when $1 \le j \ne i$, let a_j be such that $\xi = a_1 \cdots a_{i-1} a_{i+1} \cdots$. When $j \ge 0$, none of $a_1 \cdots a_{i-1} a_{i+1} \cdots a_{i+j}$ is in $Div(L) = Div(L')$. Let T be the LTS whose alphabet is $\Sigma(L) \cup \{a_i\}$ and whose graph is $\searrow\!\circ \xrightarrow{a_1} \circ \xrightarrow{a_2} \ldots$. We have $a_1 \cdots a_{i+j} \notin Div(L' \,||\, T)$ but $a_i \in Div(\,(L \,||\, T) \setminus \Sigma(L)\,) = Div(\,(L' \,||\, T) \setminus \Sigma(L')\,)$. As a consequence, $a_1 a_2 \cdots \in Inf(L' \,||\, T)$, $\xi \in Inf(L')$, and $\xi \in eanI(L')$. $\square$

6.2. **Lower sub-region.** In this subsection we survey the part of the current region that is below the dashed grey line in Fig. 6.

Thanks to the next theorem, all congruences in this and the next section preserve $minD$.

Theorem 16. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and $\searrow\!\circ \not\cong \tau \circlearrowright\!\!\downarrow\!\circ \xrightarrow{\tau} \circ$, then "$\cong$" preserves $minD$.

Proof. To derive a contradiction, let $L \cong L'$ and $\sigma = a_1 \cdots a_n \in \mathit{minD}(L) \setminus \mathit{minD}(L')$. If there is an $i < n$ such that $a_1 \cdots a_i \in \mathit{minD}(L')$, then swap the roles of L and L', and use $a_1 \cdots a_i$ instead of σ. Now no prefix of σ is in $\mathit{minD}(L')$.

If $\Sigma(L) \neq \Sigma(L')$, then Theorem 1 yields $\searrow\!\circ \cong \tau\circlearrowright\!\overset{\downarrow}{\circ}\xrightarrow{\tau}\circ$. Otherwise, let T_σ be the LTS whose graph is $\searrow\!\circ\xrightarrow{a_1}\circ\xrightarrow{a_2}\cdots\xrightarrow{a_n}\circ$ and alphabet is $\Sigma(L)$. We have $(L \mid\mid T_\sigma) \setminus \Sigma(L) \doteq \tau\circlearrowright\!\overset{\downarrow}{\circ}\xrightarrow{\tau}\circ$ or $(L \mid\mid T_\sigma) \setminus \Sigma(L) \doteq \searrow\!\circ\circlearrowleft\tau$. Furthermore, $(L' \mid\mid T_\sigma) \setminus \Sigma(L) \doteq \searrow\!\circ$. These imply $\searrow\!\circ \cong \tau\circlearrowright\!\overset{\downarrow}{\circ}\xrightarrow{\tau}\circ$ or $\searrow\!\circ \cong \searrow\!\circ\circlearrowleft\tau$. If $\searrow\!\circ \cong \searrow\!\circ\circlearrowleft\tau$, then $\searrow\!\circ\circlearrowleft\tau \doteq \tau\circlearrowright\!\overset{\downarrow}{\circ}\xrightarrow{\tau}\circ \mid\mid \searrow\!\circ\circlearrowleft\tau \cong \tau\circlearrowright\!\overset{\downarrow}{\circ}\xrightarrow{\tau}\circ \mid\mid \searrow\!\circ \equiv \tau\circlearrowright\!\overset{\downarrow}{\circ}\xrightarrow{\tau}\circ$, so $\searrow\!\circ \cong \tau\circlearrowright\!\overset{\downarrow}{\circ}\xrightarrow{\tau}\circ$. All cases contradict the assumption $\searrow\!\circ \not\cong \tau\circlearrowright\!\overset{\downarrow}{\circ}\xrightarrow{\tau}\circ$. □

By Lemmas 12 and 13, all congruences in this and the next section also preserve Σ, anT, and anI. Furthermore, in this section also Theorem 8 is applicable. So Sf must be added to the semantics. Doing so yields a congruence (proof skipped). After adding Sf, anT can be removed because $\mathit{anT}(L) = \mathit{Sf}^{\,\mathit{Tr}}(L) \setminus \mathit{extT}(L)$. Thus the weakest congruence in this section is induced by Σ, Sf, minD, and anI.

Adding Tr to this also yields a congruence. The next theorem says that it is the next congruence. We will need the construction in the proof of the theorem also in Section 7, so we isolate it in a lemma.

Lemma 17. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and "$\cong$" preserves minD but not Tr, then for every LTS L there is an LTS $f(L)$ such that $L \cong f(L)$, $\mathit{Tr}(f(L)) = \mathit{anT}(L) \cup \mathit{extT}(L)$, $\mathit{Sf}(f(L)) = \mathit{Sf}(L)$, $\mathit{Div}(f(L)) = \mathit{extT}(L)$, and $\mathit{Inf}(f(L)) = \mathit{anI}(L) \cup \mathit{extI}(L)$.

Proof. By Theorem 1, "$\cong$" preserves Σ. Let $M_1 \cong M_2$, $\sigma \in \mathit{Tr}(M_1) \setminus \mathit{Tr}(M_2)$, $b_1 \cdots b_n = \sigma^{[1]}$, and $c = 1^{[0]}$. Let $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$. For any LTS L, let $\Sigma_L = \Sigma(L)$, and let $g(L)$ be the LTS that is obtained as follows: the label of every visible transition is transformed from a to $a^{[2]}$, and a c-transition to the initial state of Fig. 3 is added to every divergent state. In Fig. 3, $\{a_1, \ldots, a_m\} = \Sigma_L^{[2]}$. The alphabet of $g(L)$ is $\{c\} \cup \Sigma_M^{[1]} \cup \Sigma_L^{[2]}$.

When $i \in \{1, 2\}$, let

$$f_i(L) \;=\; \lfloor\, (g(L) \mid\mid c.\lceil M_i \rceil^{[1]}) \setminus (\{c\} \cup \Sigma_M^{[1]})\, \rfloor_{[2]} \;.$$

By construction, $f_i(L)$ can do everything that L can do, and also try to hiddenly execute $c\sigma^{[1]}$. Attempts to execute $c\sigma^{[1]}$ start at divergent states and, thanks to the τ-loops in Fig. 3, do not lead to stable states. Thus $\mathit{Sf}(f_1(L)) = \mathit{Sf}(f_2(L)) = \mathit{Sf}(L)$. Because $f_2(L)$ cannot execute $c\sigma^{[1]}$ completely, $f_2(L) \doteq L$. On the other hand, $f_1(L)$ can, so $\mathit{Tr}(f_1(L)) = \mathit{anT}(L) \cup \mathit{extT}(L)$, $\mathit{Div}(f_1(L)) = \mathit{extT}(L)$, and $\mathit{Inf}(f_1(L)) = \mathit{anI}(L) \cup \mathit{extI}(L)$. We have $L \doteq f_2(L) \cong f_1(L)$. Therefore, f_1 qualifies as the f of the claim. □

Theorem 18. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and "$\cong$" preserves Sf and minD but not Tr, then "$\cong$" is the equivalence induced by Σ, Sf, minD, and anI.

Proof. Lemma 13 implies that "$\cong$" preserves Σ and anI. Because $\mathit{extT}(L)$ and $\mathit{extI}(L)$ are functions of $\Sigma(L)$ and $\mathit{minD}(L)$, the f of Lemma 17 qualifies as the f of Lemma 3 with $k = 3$, $X_1 = \mathit{Sf}$, $X_2 = \mathit{minD}$, and $X_3 = \mathit{anI}$. □

6.3. **Unambiguation of LTSs.** In this subsection we motivate and present two functions, called Una and PD, that transform any LTS to a bisimilar LTS that has some useful property.

To continue the survey, we need a construction that preserves anI but not Inf. It will block infinite traces after a minimal divergence trace, while not affecting them before a minimal divergence trace. Blocking does not have the desired effect unless *all* executions

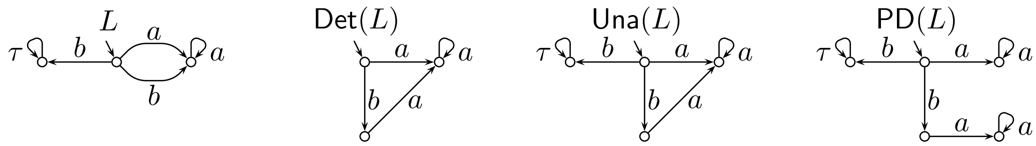


Figure 7: An example of L, $\mathsf{Det}(L)$, $\mathsf{Una}(L)$, and $\mathsf{PD}(L)$.

of each minimal divergence trace switch it on. Forcing the execution of the switch at every divergent state does not suffice, because the same trace may have two executions, one leading to a divergent and the other to a nondivergent state. This is exemplified by the trace b of the L in Fig. 7. Even if we knew that this is the case with some nondivergent state, we could not blindly implement the switch there, because it may also be reachable via another, always-nondivergent trace. An example is the trace a in the figure.

To cope with this problem, we define a function Una that, given an LTS, yields a bisimilar LTS where different traces lead to the same state only if they have the same futures. This is obtained by keeping track, in a new component of the state, of the set of original states that can be reached via the trace that has been executed so far. To do that, we first define the *determinization* of L as the LTS

$$\begin{array}{rcl} \mathsf{Det}(L) & := & (S_\mathsf{D}, \Sigma, \Delta_\mathsf{D}, \hat{s}_\mathsf{D}), \text{ where} \\ S_\sigma & = & \{s \mid \hat{s} =\sigma\Rightarrow s\} \\ S_\mathsf{D} & = & \{S_\sigma \mid \sigma \in \mathit{Tr}(L)\} \\ \Delta_\mathsf{D} & = & \{(S_\sigma, a, S_{\sigma a}) \mid a \neq \tau \wedge \sigma a \in \mathit{Tr}(L)\} \\ \hat{s}_\mathsf{D} & = & S_\varepsilon \end{array}$$

Lemma 19. If $\sigma \in \mathit{Tr}(L)$, then $\hat{s}_\mathsf{D} =\sigma\Rightarrow S_\sigma$. If $\hat{s}_\mathsf{D} =\sigma\Rightarrow s_\mathsf{D}$, then $s_\mathsf{D} = S_\sigma$ and $\sigma \in \mathit{Tr}(L)$.

Proof. We prove the first claim by induction. Clearly $\hat{s}_\mathsf{D} =\varepsilon\Rightarrow \hat{s}_\mathsf{D} = S_\varepsilon$. If $\sigma a \in \mathit{Tr}(L)$, then $a \neq \tau$ and $(S_\sigma, a, S_{\sigma a}) \in \Delta_\mathsf{D}$. By the induction assumption $\hat{s}_\mathsf{D} =\sigma\Rightarrow S_\sigma$, yielding $\hat{s}_\mathsf{D} =\sigma a\Rightarrow S_{\sigma a}$.

Also the second claim is proven by induction. The definition of Δ_D constructs no τ-transitions, so if $\hat{s}_\mathsf{D} =\varepsilon\Rightarrow s_\mathsf{D}$, then $s_\mathsf{D} = \hat{s}_\mathsf{D} = S_\varepsilon$. Trivially $\varepsilon \in \mathit{Tr}(L)$. If $\hat{s}_\mathsf{D} =\sigma a\Rightarrow s_\mathsf{D}$, then consider the last transition along the path. By the definition of Δ_D, it is of the form $(S_\rho, a, S_{\rho a})$, where $\rho a \in \mathit{Tr}(L)$, $\hat{s}_\mathsf{D} =\sigma\Rightarrow S_\rho$, and $S_{\rho a} = s_\mathsf{D}$. By the induction assumption $S_\rho = S_\sigma$. We get $s_\mathsf{D} = S_{\rho a} = \{s \mid \exists s' \in S_\rho : s' =a\Rightarrow s\} = \{s \mid \exists s' \in S_\sigma : s' =a\Rightarrow s\} = S_{\sigma a}$. Because $\rho a \in \mathit{Tr}(L)$, we have $\emptyset \neq S_{\rho a} = S_{\sigma a}$, so $\sigma a \in \mathit{Tr}(L)$. □

Then we define the *unambiguation* of L as

$$\mathsf{Una}(L) \quad := \quad L \,||\, \mathsf{Det}(L) \ .$$

Lemma 20. $\mathsf{Una}(L) \equiv L$, that is, $\mathsf{Una}(L)$ is bisimilar with L.

Proof. Let $\mathsf{Una}(L) = (S_\mathsf{U}, \Sigma_\mathsf{U}, \Delta_\mathsf{U}, \hat{s}_\mathsf{U})$. The states of $\mathsf{Una}(L)$ are of the form $s_\mathsf{U} = (s, S_\sigma)$. Let "$\sim$" $\subseteq S \times S_\mathsf{U}$ be defined by $s \sim (s', S_\sigma)$ if and only if $\hat{s} =\sigma\Rightarrow s = s'$. We now show that "$\sim$" is a bisimulation. "(1)", etc., refer to the numbers in the definition on p. 3.

(1) Clearly $\Sigma(\mathsf{Una}(L)) = \Sigma_\mathsf{U} = \Sigma \cup \Sigma = \Sigma = \Sigma(L)$.
(2) We have $\hat{s}_\mathsf{U} = (\hat{s}, \hat{s}_\mathsf{D}) = (\hat{s}, S_\varepsilon)$ and $\hat{s} =\varepsilon\Rightarrow \hat{s}$, so $\hat{s} \sim \hat{s}_\mathsf{U}$.
(3) Let $s \sim (s, S_\sigma)$, that is, $\hat{s} =\sigma\Rightarrow s$.

(3a) If $(s, \tau, s') \in \Delta$, then $((s, S_\sigma), \tau, (s', S_\sigma)) \in \Delta_{\mathsf{U}}$ and $\hat{s} =\sigma\Rightarrow s'$, yielding $s' \sim (s', S_\sigma)$. If $(s, a, s') \in \Delta$ where $a \in \Sigma$, then $\hat{s} =\sigma a\Rightarrow s'$. The definition of Δ_{D} yields $(S_\sigma, a, S_{\sigma a}) \in \Delta_{\mathsf{D}}$, implying $((s, S_\sigma), a, (s', S_{\sigma a})) \in \Delta_{\mathsf{U}}$. We have $s' \sim (s', S_{\sigma a})$.

(3b) If $((s, S_\sigma), \tau, (s', s'_{\mathsf{D}})) \in \Delta_{\mathsf{U}}$, then by the definitions of "$||$" and Δ_{D} we have $(s, \tau, s') \in \Delta$ and $s'_{\mathsf{D}} = S_\sigma$. Furthermore, $\hat{s} =\sigma\Rightarrow s'$. So $s' \sim (s', s'_{\mathsf{D}})$. If $((s, S_\sigma), a, (s', s'_{\mathsf{D}})) \in \Delta_{\mathsf{U}}$ where $a \in \Sigma$, then $(s, a, s') \in \Delta$. It implies $\hat{s} =\sigma a\Rightarrow s'$. We also have $(S_\sigma, a, s'_{\mathsf{D}}) \in \Delta_{\mathsf{D}}$, yielding $\sigma a \in \mathit{Tr}(\mathsf{Det}(L))$ and $s'_{\mathsf{D}} = S_{\sigma a}$ by Lemma 19. Again $s' \sim (s', s'_{\mathsf{D}})$. □

We say that a state of $\mathsf{Una}(L)$ is *potentially divergent* if it can be reached via a divergence trace, and *certainly nondivergent* otherwise. These phrases do not actually refer to the properties of the state but to the properties of the traces that lead to it. The essential useful property of $\mathsf{Una}(L)$ is stated in the following lemma.

Lemma 21. If state s_{U} of $\mathsf{Una}(L)$ is potentially divergent, then all traces that lead to it belong to $\mathit{Div}(L)$. If state s_{U} of $\mathsf{Una}(L)$ is certainly nondivergent, then no trace that leads to it belongs to $\mathit{Div}(L)$.

Proof. If $\hat{s}_{\mathsf{U}} =\sigma\Rightarrow s_{\mathsf{U}}$, then s_{U} is of the form (s, s_{D}), where $\hat{s} =\sigma\Rightarrow s$ and $\hat{s}_{\mathsf{D}} =\sigma\Rightarrow s_{\mathsf{D}}$. By Lemma 19, $s_{\mathsf{D}} = S_\sigma$. If also $\hat{s}_{\mathsf{U}} =\rho\Rightarrow s_{\mathsf{U}}$, then $S_\rho = s_{\mathsf{D}} = S_\sigma$. If $\sigma \in \mathit{Div}(L)$, then there is an $s' \in S_\sigma = S_\rho$ such that $s' -\tau^\omega\rightarrow$, implying $\rho \in \mathit{Div}(L)$. Therefore, either none or all of the traces that lead to s_{U} are divergence traces. □

In Fig. 7, the rightmost state of L has been split to two states in $\mathsf{Una}(L)$, a certainly nondivergent one led to by a and a potentially divergent one led to by b.

Then we define a function PD that makes the following property hold while preserving bisimilarity: for every state s, either no or all traces that lead to s have a divergence trace as a prefix. This is obtained by adding a component to $\mathsf{Una}(L)$ that remembers if the execution has gone through a divergence trace. Formally, by $\mathsf{PD}(L)$ we mean the LTS $(S_{\mathsf{P}}, \Sigma, \Delta_{\mathsf{P}}, \hat{s}_{\mathsf{P}})$ that is obtained as follows. Let $[\sigma] = \mathsf{pre}$ if $\sigma \in \mathit{anT}(L)$ and $[\sigma] = \mathsf{post}$ otherwise. Let $\sigma_\tau = \sigma$ and $\sigma_a = \sigma a$ if $a \in \Sigma$. First L is replaced by $\mathsf{Una}(L) = (S_{\mathsf{U}}, \Sigma, \Delta_{\mathsf{U}}, \hat{s}_{\mathsf{U}})$. Then let

$$\begin{aligned} S_{\mathsf{P}} &= \{(s_{\mathsf{U}}, [\sigma]) \mid \hat{s}_{\mathsf{U}} =\sigma\Rightarrow s_{\mathsf{U}}\} \\ \Delta_{\mathsf{P}} &= \{((s_{\mathsf{U}}, [\sigma]), a, (s'_{\mathsf{U}}, [\sigma_a])) \mid \hat{s}_{\mathsf{U}} =\sigma\Rightarrow s_{\mathsf{U}} \wedge (s_{\mathsf{U}}, a, s'_{\mathsf{U}}) \in \Delta_{\mathsf{U}}\} \\ \hat{s}_{\mathsf{P}} &= (\hat{s}_{\mathsf{U}}, [\varepsilon]) \end{aligned}$$

We say that (s_{U}, x) is *pre-divergent* if $x = \mathsf{pre}$ and *post-divergent* otherwise.

Lemma 22. We have $\mathsf{PD}(L) \equiv L$. If state s_{P} of $\mathsf{PD}(L)$ is pre-divergent, then all traces that lead to it belong to $\mathit{anT}(L)$. If state s_{P} of $\mathsf{PD}(L)$ is post-divergent, then no trace that leads to it belongs to $\mathit{anT}(L)$.

Proof. We have $\mathsf{PD}(L) \equiv \mathsf{Una}(L) \equiv L$, because the relation $(s_{\mathsf{U}}, [\sigma]) \sim s'_{\mathsf{U}} \Leftrightarrow s_{\mathsf{U}} = s'_{\mathsf{U}}$ is a bisimulation between S_{P} and S_{U}.

If $[\sigma_a] = \mathsf{pre}$, then $\sigma_a \in \mathit{anT}(L)$, implying $\sigma \in \mathit{anT}(L)$ and $[\sigma] = \mathsf{pre}$. Thus $\mathsf{PD}(L)$ has no transitions from post-divergent to pre-divergent states.

Let $\hat{s}_{\mathsf{P}} =\rho\Rightarrow (s_{\mathsf{U}}, x)$ and $\rho \in \mathit{Div}(L)$. Because $(s_{\mathsf{U}}, x) \in S_{\mathsf{P}}$, there is a σ such that $\hat{s}_{\mathsf{U}} =\sigma\Rightarrow s_{\mathsf{U}}$ and $x = [\sigma]$. Because $\rho \in \mathit{Div}(L)$, s_{U} is potentially divergent. By Lemma 21, all traces that lead to it are divergence traces. That includes σ. Thus $x = \mathsf{post}$. As a consequence, each divergence trace only leads to post-divergent states. By the first result in this proof, the same holds for each trace that has a divergence trace as a prefix.

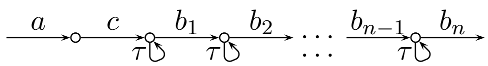


Figure 8: An LTS fragment for detecting the divergence trace $b_1 b_2 \cdots b_n$.

If an execution of $\mathsf{PD}(L)$ leads to a post-divergent state, then $\hat{s}_\mathsf{P}$ is post-divergent or the execution contains a transition of the form $((s_\mathsf{U}, \mathsf{pre}), a, (s'_\mathsf{U}, \mathsf{post}))$. In the first case, $[\varepsilon] = \mathsf{post}$, so $\varepsilon \in \mathit{Div}(L)$. In the second case, by the definition of Δ_P, there is a σ such that $\hat{s}_\mathsf{U} =\sigma\Rightarrow s_\mathsf{U}$, $\sigma \in \mathit{anT}(L)$, and $\sigma_a \notin \mathit{anT}(L)$. This implies $\sigma a \in \mathit{Div}(L)$. So s'_U is potentially divergent and all traces that lead to it are divergence traces. As a consequence, each post-divergent state has a divergence trace in each of its histories. $\square$

In Fig. 7, the rightmost state of $\mathsf{Una}(L)$ has been split to two states in $\mathsf{PD}(L)$, one such that all traces leading to it start with the only divergence trace b, and another such that no trace leading to it starts with b.

6.4. **Upper sub-region.** In this subsection we survey the rest of the current region.

Armed with PD, we can attack the case where Tr, Sf, and minD are preserved, but Div and Inf are not. This time there is no unique next congruence, but two. Therefore, the proof consists of two parts, where the first throws away information on divergence traces that are not minimal, and the second on infinite traces that are not always-nondivergent. Again, to reuse the construction in Section 7, we present it as a lemma that does not assume that Sf is preserved.

Lemma 23. Assume that "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and "$\cong$" preserves Tr and minD but not Div.

(a) For every LTS L there is an LTS $f(L)$ such that $L \cong f(L)$, $\mathit{Sf}(f(L)) = \mathit{Sf}(L)$, $\mathit{Div}(f(L)) = \mathit{Tr}(L) \cap \mathit{extT}(L)$, and $\mathit{Inf}(f(L)) = \mathit{Inf}(L)$.

(a) If "$\cong$" does not preserve Inf, then for every LTS L there is an LTS $f(L)$ such that $L \cong f(L)$, $\mathit{Sf}(f(L)) = \mathit{Sf}(L)$, $\mathit{Div}(f(L)) = \mathit{Tr}(L) \cap \mathit{extT}(L)$, and $\mathit{Inf}(f(L)) = \mathit{anI}(L)$.

Proof. Let $M_1 \cong M_2$, $\sigma \in \mathit{Div}(M_1) \setminus \mathit{Div}(M_2)$, $b_1 \cdots b_n = \sigma^{[1]}$, $c = 1^{[0]}$, and $d = 2^{[0]}$. By Theorem 1, "$\cong$" preserves Σ, so we may let $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$. For any LTS L, let $\Sigma_L = \Sigma(L)$ and let $g(L)$ be the LTS that is obtained as follows. First L is replaced by $\mathsf{PD}(\lceil L \rceil^{[2]})$. If $\hat{s}_\mathsf{P}$ is pre-divergent, then it is the new initial state, and each transition (s, a, s') where s is pre-divergent and s' is post-divergent is replaced by a copy of the LTS fragment shown in Fig. 8. Otherwise a copy of Fig. 8 is added such that its a-transition is left out, the start state of the c-transition is the new initial state, and the LTS fragment leads to $\hat{s}_\mathsf{P}$. The alphabet of $g(L)$ is $\{c\} \cup \Sigma_M^{[1]} \cup \Sigma_L^{[2]}$. When completing a minimal divergence trace of $\lceil L \rceil^{[2]}$, $g(L)$ executes $c\sigma^{[1]}$ before continuing, but otherwise it behaves like $\lceil L \rceil^{[2]}$.

Later, in the proof of claim (b), we will introduce Σ_N, N'_1, and N'_2. To have a place for them in our construction, we now let $N'_0 = \searrow\!\circ$ (with $\Sigma(N'_0) = \emptyset$). When $i \in \{1, 2\}$ and $j \in \{0, 1, 2\}$, let $M'_i = c.\lceil M_i \sqcap M_2 \rceil^{[1]}$ and

$$f_{i,j}(L) \;=\; \lfloor\, (\, g(L) \,||\, M'_i \;\; ||\, N'_j \,) \setminus (\{c, d\} \cup \Sigma_M^{[1]} \cup \Sigma_N^{[3]}) \,\rfloor_{[2]} \;.$$

Clearly N'_0 has no effect on the behaviour. With N'_0, independently of what Σ_N is, also the hiding with $\Sigma_N^{[3]}$ has no effect.

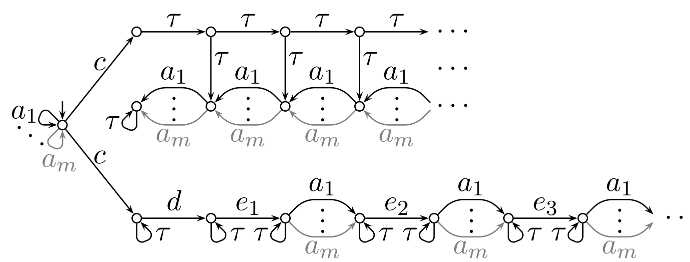


Figure 9: A switchable LTS for detecting the infinite trace $e_1e_2\cdots$.

We show now that $L \doteq f_{2,0}(L)$. Before completing any minimal divergence trace, $f_{2,0}(L)$ behaves like L. When $g(L)$ executes c, one of the two copies of M_2 in M_2' is switched on. Then $g(L)$ tries to execute $\sigma^{[1]}$. If it fails because M_2 blocks it, then $f_{2,0}(L)$ diverges due to the τ-loops in Fig. 8. That is still equivalent to L, because the trace that has been executed is a minimal divergence trace. For the same reason it is okay if M_2 diverges before completing σ. The execution of σ may also succeed, because $\sigma \in \mathit{Div}(M_1) \subseteq \mathit{Tr}(M_1) = \mathit{Tr}(M_2)$. In that case, because $\sigma \notin \mathit{Div}(M_2)$, M_2 is left in a nondivergent state, having no effect on the further behaviour. So $f_{2,0}(L)$ continues like L.

Because M_1' has a copy of both M_1 and M_2, $f_{1,0}(L)$ behaves otherwise like $f_{2,0}(L)$, but it has additional behaviour caused by M_1 starting in M_1', executing σ completely, and diverging. In that case, every subsequent state of $f_{1,0}(L)$ is divergent. Thus $L \doteq f_{2,0}(L) \cong f_{1,0}(L)$, $\mathit{Tr}(f_{1,0}(L)) = \mathit{Tr}(L)$, $\mathit{Sf}(f_{1,0}(L)) = \mathit{Sf}(L)$, $\mathit{minD}(f_{1,0}(L)) = \mathit{minD}(L)$, $\mathit{Div}(f_{1,0}(L)) = \mathit{Tr}(L) \cap \mathit{extT}(L)$, $\mathit{anI}(f_{1,0}(L)) = \mathit{anI}(L)$, and $\mathit{Inf}(f_{1,0}(L)) = \mathit{Inf}(L)$. As a consequence, $f_{1,0}$ qualifies as the f of claim (a).

In the case of claim (b), there are N_1, N_2, and ξ such that $N_1 \cong N_2$ and $\xi \in \mathit{Inf}(N_1) \setminus \mathit{Inf}(N_2)$. Let $e_1e_2\cdots = \xi^{[3]}$, $\Sigma_N = \Sigma(N_1) = \Sigma(N_2)$, and $\{a_1, a_2, \ldots, a_m\} = \Sigma_L^{[2]}$. When $j \in \{1,2\}$, let $N_j' = T_\xi \mid\mid d.\lceil N_j \rceil^{[3]}$, where T_ξ is the LTS in Fig. 9 with the alphabet $\{c,d\} \cup \Sigma_L^{[2]} \cup \Sigma_N^{[3]}$.

If $j \in \{1,2\}$, c makes T_ξ enter one of its two branches. Its initial state and upper branch can parallel any finite execution of $g(L)$. Because T_ξ never refuses any other subset of $\Sigma_L^{[2]}$ than $\emptyset$, and because of the stable states initially and in the upper branch, $\mathit{Sf}(f_{1,j}(L)) = \mathit{Sf}(f_{1,0}(L))$. Furthermore, $\mathit{Div}(f_{1,j}(L)) = \mathit{Div}(f_{1,0}(L))$, because T_ξ cannot diverge before executing c, and all traces that involve the execution of c are in $\mathit{Div}(f_{1,0}(L))$.

The upper branch of T_ξ does not yield infinite traces. In its lower branch T_ξ switches N_j on by executing d. Thanks to the initial state of T_ξ and because N_2 cannot execute ξ, we have $\mathit{Inf}(f_{1,2}(L)) = \mathit{anI}(f_{1,0}(L))$. Because N_1 can execute ξ, we have $\mathit{Inf}(f_{1,1}(L)) = \mathit{Inf}(f_{1,0}(L))$. We get $f_{1,0}(L) \doteq f_{1,1}(L) \cong f_{1,2}(L)$. So $f_{1,2}$ qualifies as the f of claim (b). □

Theorem 24. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and "$\cong$" preserves Tr, Sf, and minD but neither Div nor Inf, then "$\cong$" is the equivalence induced by Σ, Tr, Sf, minD, and anI.

Proof. By Lemma 13, "$\cong$" preserves Σ and anI. The f of Lemma 23(b) qualifies as the f of Lemma 3. □

We have now two directions to go: one where Inf is preserved and another where Div is preserved. Given the work we have done already, the former is easy.

Theorem 25. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and "$\cong$" preserves Sf, minD, and Inf but not Div, then "$\cong$" is the equivalence induced by Σ, Tr, Sf, minD, and Inf.

Proof. By Lemma 2, "$\cong$" preserves Σ and Tr. The f of Lemma 23(a) qualifies as the f of Lemma 3. □

We still have the case where Div is preserved but Inf is not.

Lemma 26. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and "$\cong$" preserves Div but not aenI, then for every LTS L there is an LTS $f(L)$ such that $L \cong f(L)$, $\mathit{Sf}(f(L)) = \mathit{Sf}(L)$, $\mathit{Div}(f(L)) = \mathit{Div}(L)$, and $\mathit{Inf}(f(L)) = \mathit{eanI}(L)$.

Proof. Let $M_1 \cong M_2$ and $\xi \in \mathit{aenI}(M_1) \setminus \mathit{aenI}(M_2)$. By Theorem 1, "$\cong$" preserves Σ. Let $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$, $c = 0^{[0]}$, $c_1 = 1^{[0]}$, and $c_2 = 2^{[0]}$. Because "$\cong$" preserves Div, M_1 and M_2 agree on which prefixes of ξ are divergence traces. Infinitely many of them are not, by the definition of aenI. So non-empty σ_1, σ_2, σ_3, $\ldots$ exist such that $\sigma_1\sigma_2\sigma_3\cdots = \xi^{[1]}$ and σ_1, $\sigma_1\sigma_2$, $\sigma_1\sigma_2\sigma_3$, $\ldots$ are not divergence traces. Let T_ξ be the LTS whose alphabet is $\{c, c_1, c_2\} \cup \Sigma_M^{[1]}$ and whose graph is

$$\searrow\circ \xrightarrow{c_1} \circ \xrightarrow{c} \circ \overset{\sigma_1}{\rightsquigarrow} \circ \xrightarrow{c_2} \circ \xrightarrow{c_1} \circ \overset{\sigma_2}{\rightsquigarrow} \circ \xrightarrow{c_2} \circ \xrightarrow{c_1} \circ \overset{\sigma_3}{\rightsquigarrow} \cdots .$$

For any LTS L, let $g(L)$ be the LTS that is obtained as follows. First L is replaced by $\mathsf{Una}(\lceil L \rceil^{[2]})$. Then each transition whose label a is visible and which ends in a potentially divergent state is replaced by $\xrightarrow{a} \circ \xrightarrow{c_1} \overset{\tau}{\circlearrowleft}\!\circ \xrightarrow{c_2}$. The alphabet of the result is $\{c_1, c_2\} \cup \Sigma_L^{[2]}$, where $\Sigma_L = \Sigma(L)$. When $i \in \{1, 2\}$, let

$$f_i(L) \;=\; \lfloor\, (\, g(L) \,||\, T_\xi \,||\, c.\lceil M_i \rceil^{[1]}\,) \setminus (\{c, c_1, c_2\} \cup \Sigma_M^{[1]})\, \rfloor_{[2]} \;.$$

Each time when $g(L)$ is about to enter a potentially divergent state, it executes c_1. This makes T_ξ move one step and then let $c.\lceil M_i \rceil^{[1]}$ try to execute up to a nondivergent state. If it succeeds, T_ξ lets $g(L)$ continue by executing c_2. In the opposite case, $g(L)$ is trapped in the τ-loop between c_1 and c_2.

The LTS M_1 has every prefix of ξ as its trace. By Lemma 14, "$\cong$" preserves Tr. So both $\lceil M_1 \rceil^{[1]}$ and $\lceil M_2 \rceil^{[1]}$ may succeed in executing $\sigma_1\sigma_2\cdots\sigma_i$ for any i. This implies $\mathit{Tr}(f_1(L)) = \mathit{Tr}(f_2(L)) = \mathit{Tr}(L)$. Clearly $g(L)$ mimics the divergence traces of L. When M_1 or M_2 diverges, $g(L)$ is in a τ-loop and the trace that has been executed is a divergence trace. Thus $\mathit{Div}(f_1(L)) = \mathit{Div}(f_2(L)) = \mathit{Div}(L)$.

When $g(L)$ is in a stable state (other than the start states of c_1), then $c.\lceil M_1 \rceil^{[1]}$ and $c.\lceil M_2 \rceil^{[1]}$ do not diverge, so $\mathit{Sf}(f_1(L)) = \mathit{Sf}(f_2(L)) = \mathit{Sf}(L)$. Because M_2 does but M_1 does not necessarily prevent $g(L)$ from infinitely many times continuing with c_2 after a divergence trace, we have $\mathit{Inf}(f_1(L)) = \mathit{Inf}(L)$ but $\mathit{Inf}(f_2(L)) = \mathit{eanI}(L)$. So $L \doteq f_1(L) \cong f_2(L)$. □

Theorem 27. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and "$\cong$" preserves Sf and Div but not aenI, then "$\cong$" is the equivalence induced by Σ, Sf, Div, and eanI.

Proof. By Lemma 15, "$\cong$" preserves Σ and eanI. The f of Lemma 26 qualifies as the f of Lemma 3. □

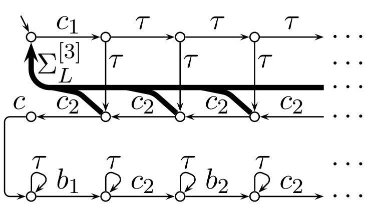


Figure 10: An LTS for detecting an infinite trace with only finitely many nondivergent prefixes. The thick arrows with $\Sigma_L^{[3]}$ denote that there is a transition from each start state of the thick arrows to their common end state for each $a \in \Sigma_L^{[3]}$.

Lemma 28. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and "$\cong$" preserves *Div* and *aenI* but not *Inf*, then for every LTS L there is an LTS $f(L)$ such that $L \cong f(L)$, $Sf(f(L)) = Sf(L)$, $Div(f(L)) = Div(L)$, and $Inf(f(L)) = aenI(L)$.

Proof. For the purpose of this proof, we define *eventually-always-divergent* infinite traces as $eadI(L) = Inf(L) \setminus aenI(L)$. Let $M_1 \cong M_2$ and $\xi \in Inf(M_1) \setminus Inf(M_2)$. By Theorem 1, "$\cong$" preserves Σ. Let $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$, $c = 0^{[0]}$, $c_1 = 1^{[0]}$, and $c_2 = 2^{[0]}$. Because "$\cong$" preserves *aenI*, $\xi \in eadI(M_1)$. Because "$\cong$" preserves *Div*, M_1 and M_2 agree on which prefixes of ξ are divergence traces. From some point on all of them are, because $\xi \in eadI(M_1)$.

For any LTS L, let $\Sigma_L = \Sigma(L)$, and let $g(L)$ be obtained as follows. Each transition of $\mathsf{Una}(L)$ whose label a is visible is replaced by

$\xrightarrow{a^{[2]}}\circ\xrightarrow{c_1}$, if it starts in a certainly nondiv. and ends in a potentially divergent state;
$\xrightarrow{a^{[2]}}\circ\xrightarrow{c_2}$, if it starts and ends in a potentially divergent state;
$\xrightarrow{a^{[3]}}$, if it starts in a potentially divergent and ends in a certainly nondiv. state;
$\xrightarrow{a^{[2]}}$, if it starts and ends in a certainly nondivergent state.

If the initial state of $\mathsf{Una}(L)$ is potentially divergent, then a c_1-transition is added to its front. The alphabet of $g(L)$ is $\{c_1, c_2\} \cup \Sigma_L^{[2]} \cup \Sigma_L^{[3]}$.

Let $b_1 b_2 \cdots = \xi^{[1]}$. Let T_ξ be the LTS whose alphabet is $\{c, c_1, c_2\} \cup \Sigma_M^{[1]} \cup \Sigma_L^{[3]}$ and whose graph is in Fig. 10. When $i \in \{1, 2\}$, let

$$f_i(L) \ = \ \big((g(L) \,||\, T_\xi \,||\, c.\lceil M_i \rceil^{[1]}) \setminus (\{c, c_1, c_2\} \cup \Sigma_M^{[1]}) \big) \Phi \ ,$$

where Φ renames each $a^{[2]}$ and each $a^{[3]}$ to a.

While $g(L)$ traverses among certainly nondivergent states, $f_1(L)$ and $f_2(L)$ behave like L. When $g(L)$ enters a potentially divergent state, T_ξ prepares for an arbitrary finite number of transitions between potentially divergent states. The divergence of T_ξ is not a problem, because the trace that has been executed is a divergence trace. As long as T_ξ is in its middle row excluding its leftmost state, $g(L)$ can execute transitions at will. These states of T_ξ are stable and offer all actions in $\Sigma(g(L)) \cap \Sigma(T_\xi)$ except c_1 that also $g(L)$ refuses, so *Sf* is preserved. If $g(L)$ enters a certainly nondivergent state, then T_ξ goes back to its initial state. As a consequence, $f_1(L)$ and $f_2(L)$ have at least the same stable failures, divergence traces, and always-eventually-nondivergent infinite traces as L, and no extra stable failures, divergence traces, or infinite traces have so far been found.

If $g(L)$ executes more transitions between potentially divergent states than T_ξ has been prepared for, T_ξ reaches the leftmost state of its middle row. Then it executes c,

Table 1: All congruences when no two of deadlock, livelock, and bothlock are equivalent

preserves	does not preserve	induced by	theorem
Sf, minD	Tr	Σ, Sf, minD, anI	18
Tr, Sf, minD	Div, Inf	Σ, Tr, Sf, minD, anI	24
Sf, minD, Inf	Div	Σ, Tr, Sf, minD, Inf	25
Sf, Div	aenI	Σ, Sf, Div, eanI	27
Sf, Div, aenI	Inf	Σ, Sf, Div, aenI	29
Sf, Div, Inf		Σ, Sf, Div, Inf	

switching M_1 or M_2 on. From then on all states are divergent and $g(L)$ is prevented from leaving potentially divergent states, so no new stable failures or divergence traces are introduced. $f_2(L)$ does not introduce any new infinite traces either, while $f_1(L)$ may execute all the remaining infinite traces of L, that is, $\mathit{eadI}(L)$. So $\mathit{Sf}(f_1(L)) = \mathit{Sf}(f_2(L)) = \mathit{Sf}(L)$, $\mathit{Div}(f_1(L)) = \mathit{Div}(f_2(L)) = \mathit{Div}(L)$, $\mathit{Inf}(f_1(L)) = \mathit{Inf}(L)$, and $\mathit{Inf}(f_2(L)) = \mathit{aenI}(L)$. Clearly $L \doteq f_1(L) \cong f_2(L)$. □

Theorem 29. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and "$\cong$" preserves Sf, Div, and aenI but not Inf, then "$\cong$" is the equivalence induced by Σ, Sf, Div, and aenI.

Proof. By Theorem 1, "$\cong$" preserves Σ. The f of Lemma 28 qualifies as the f of Lemma 3. □

Both branches of reasoning have now led to congruences that preserve both Div and Inf. In this section also Sf is preserved. The equivalence induced by Σ, Sf, Div, and Inf is "$\doteq$". So "$\doteq$" is the last congruence in this section.

There are thus six congruences in this section. They are summarized in Table 1. If a congruence is implied by "$\doteq$" and preserves the sets in the first column of the table but does not preserve the sets in the second column, then it is the equivalence induced by the sets in the third column. The sets in the third column that are not in the first column of the same row must be added to meet the congruence requirement while preserving the sets in the first column. By Theorems 8 and 16, the congruence on the first row is the weakest in this section. By comparing the second column to the first column one may check that all possibilities between the first row and "$\doteq$" are covered.

7. When Deadlock Is Not Livelock Is Bothlock

In this section $\not\cong$ $\tau \cong \tau$ τ. By Theorem 16, minD is preserved also in this section. However, $\tau \cong \tau$ τ implies that Sf is not preserved. Subsection 7.1 introduces the new kinds of failures that replace Sf. The region is shown in Fig. 11. Its two lowest and the highest layer are surveyed in Subsections 7.2 and 7.3, respectively.

7.1. New kinds of failures.

In this subsection we define four new kinds of failures and briefly analyse their relation to divergence traces.

The next lemma reveals that the essence of $\tau \cong \tau$ τ is that those stable failures whose trace is a divergence trace do not matter. The function ν in the lemma throws away all information on such failures, by making $\nu(L)$ have the maximum possible set of them allowed by $\Sigma(L)$, independently of what L has.

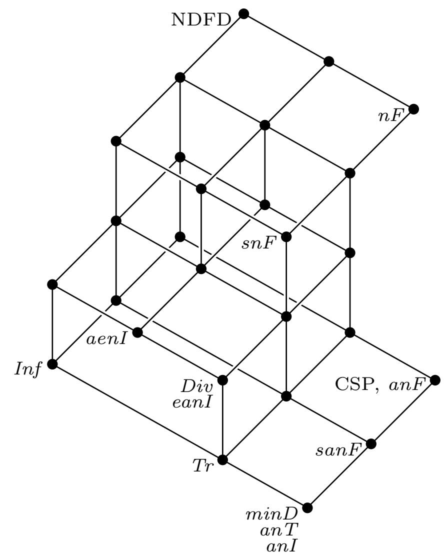


Figure 11: The congruences in Section 7 as a Hasse diagram. Names in *italics* indicate the new preserved set(s). Other names are the names of the congruences.

Lemma 30. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", and $\circ\!\circlearrowleft\tau \cong \tau\circlearrowleft\!\circ\xrightarrow{\tau}\circ$, then for every LTS L there is an LTS $\nu(L)$ such that $\nu(L) \cong L$, $\Sigma(\nu(L)) = \Sigma(L)$, $\mathit{Sf}(\nu(L)) = \mathit{Sf}(L) \cup (\mathit{Div}(L) \times 2^{\Sigma(L)})$, $\mathit{Div}(\nu(L)) = \mathit{Div}(L)$, and $\mathit{Inf}(\nu(L)) = \mathit{Inf}(L)$.

Proof. Let $M_1 = \circ\!\circlearrowleft\tau$ and $M_2 = \tau\circlearrowleft\!\circ\xrightarrow{\tau}\circ$. Let $c \notin \Sigma(L) \cup \{\tau\}$. Let $g(L)$ be the LTS that is obtained by adding, from each divergent state of L, a c-transition to a deadlock state. When $i \in \{1,2\}$, let $f_i(L) = (g(L) \,||\, c.M_i) \setminus \{c\}$. The only difference of $f_1(L)$ from L is an additional divergence where L already has a divergence, so $L \doteq f_1(L)$. On the other hand, $f_2(L)$ also has there a deadlock. Thus $f_2(L)$ has the properties promised of $\nu(L)$. $\square$

In this section we have to proceed in two dimensions. On one hand, we have to start with no information on stable failures and add it until we have all stable failures whose trace is not a divergence trace. On the other hand, for each level of information on stable failures, we have to investigate different kinds of divergence and infinite traces, like in the previous section.

We will need four new kinds of failures: *nondivergent*, *strongly nondivergent*, *always nondivergent*, and *strongly always nondivergent*.

$$\begin{array}{rcl}
\mathit{nF}(L) & := & \{(\sigma, A) \in \mathit{Sf}(L) \mid \sigma \notin \mathit{Div}(L)\} \\
\mathit{snF}(L) & := & \{(\sigma, A) \in \mathit{nF}(L) \mid \forall a \in A : \sigma a \notin \mathit{Div}(L)\} \\
\mathit{anF}(L) & := & \{(\sigma, A) \in \mathit{Sf}(L) \mid \sigma \notin \mathit{extT}(L)\} \\
\mathit{sanF}(L) & := & \{(\sigma, A) \in \mathit{anF}(L) \mid \forall a \in A : \sigma a \notin \mathit{minD}(L)\}
\end{array}$$

All these four sets $X(L)$ have the property that if $(\sigma, A) \in X(L)$, then $(\sigma, \emptyset) \in X(L)$ and $\sigma \in \mathit{Tr}(L)$. Like before, with $X^{\mathit{Tr}}(L)$ we denote the set $\{\sigma \mid (\sigma, \emptyset) \in X(L)\}$. We have the following:

$$\begin{array}{rcccl}
\mathit{nF}^{\mathit{Tr}}(L) & = & \mathit{snF}^{\mathit{Tr}}(L) & = & \mathit{Tr}(L) \setminus \mathit{Div}(L) \\
\mathit{anF}^{\mathit{Tr}}(L) & = & \mathit{sanF}^{\mathit{Tr}}(L) & = & \mathit{anT}(L)
\end{array}$$

The ν of Lemma 30 satisfies

$$\mathit{Sf}(\nu(L)) \;=\; \mathit{Sf}(L) \cup (\mathit{Div}(L) \times 2^{\Sigma(L)}) \;=\; \mathit{nF}(L) \cup (\mathit{Div}(L) \times 2^{\Sigma(L)}) \;.$$

The number of possible combinations of semantic sets is restricted a bit by the next lemma.

Lemma 31. Any congruence that preserves nF or snF also preserves Div.

Proof. By Theorem 1, it preserves Σ. Let $\sigma = a_1 \cdots a_n$. We have $\sigma \in \mathit{Div}(L)$ if and only if $\sigma \notin \mathit{nF}^{\mathit{Tr}}(L \sqcap \circ\!\xrightarrow{a_1}\!\circ\!\xrightarrow{a_2} \ldots \xrightarrow{a_n}\!\circ)$. The same proof works for snF. □

7.2. **(Strongly) always nondivergent failures.** In this subsection, we essentially repeat the analysis in Section 6 three times, with nothing, sanF, or anF in the place of Sf. Of course, we also prove that if any information on stable failures is preserved then sanF is preserved, at the next level anF or snF is preserved, and then both are preserved.

The next lemma is central in proving that if any information on stable failures is preserved, then at least sanF must be preserved.

Lemma 32. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves minD but not sanF, and $\circ\!\circlearrowleft\tau \cong \tau\circlearrowright\!\circ\!\xrightarrow{\tau}\!\circ$, then for every LTS L there is an LTS $h(L)$ such that $h(L) \cong L$, $\mathit{Sf}(h(L)) = \mathit{Tr}(L) \times 2^{\Sigma(L)}$, $\mathit{Div}(h(L)) = \mathit{Div}(L)$, and $\mathit{Inf}(h(L)) = \mathit{Inf}(L)$.

Proof. By Theorem 1, "$\cong$" preserves Σ. Let $M_1 \cong M_2$ and $(\sigma, A) \in \mathit{sanF}(M_1) \setminus \mathit{sanF}(M_2)$. Let $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$, $b_1 \cdots b_n = \sigma^{[1]}$, and $\{a_1, \ldots, a_m\} = A^{[1]}$. Let L be any LTS and $\Sigma_L = \Sigma(L)$. Let $T_{\sigma,A}$ be like in Fig. 2, except that each τ-loop is replaced by an $a^{[2]}$-loop for each $a \in \Sigma_L$, and the alphabet is $\Sigma_M^{[1]} \cup \Sigma_L^{[2]}$. When $i \in \{1, 2\}$, let

$$g(M_i) \;=\; \lfloor (T_{\sigma,A} \,||\, \lceil M_i \rceil^{[1]}) \setminus \Sigma_M^{[1]} \rfloor_{[2]} \;.$$

By the definition of sanF, $g(M_1)$ does not diverge. Because "$\cong$" preserves minD, $g(M_2)$ does not diverge. We have $\mathsf{Run}(\Sigma_L) \doteq g(M_2) \cong g(M_1) \doteq \mathsf{RD}(\Sigma_L)$, where $\mathsf{RD}(\Sigma_L)$ is obtained from $\mathsf{Run}(\Sigma_L)$ by adding a second state and a τ-transition to it from the original state (please see Fig. 1).

We have $L \equiv L \,||\, \mathsf{Run}(\Sigma_L) \cong L \,||\, \mathsf{RD}(\Sigma_L) \cong \nu(L \,||\, \mathsf{RD}(\Sigma_L))$, where ν is from Lemma 30. The LTS $L||\mathsf{RD}(\Sigma_L)$ is otherwise like L, but its stable failures are $\mathit{Sf}^{\mathit{Tr}}(L) \times 2^{\Sigma_L}$. Therefore, and given (3.1), $\nu(L \,||\, \mathsf{RD}(\Sigma_L))$ qualifies as the $h(L)$. □

We can now list the first six congruences in this section, and prove that the next ones must preserve sanF.

Table 2: The congruences of Theorem 33

preserves	does not preserve	induced by
minD	Tr, sanF	Σ, anT, minD, anI
Tr, minD	sanF, Div, Inf	Σ, Tr, minD, anI
minD, Inf	sanF, Div	Σ, Tr, minD, Inf
Div	sanF, aenI	Σ, Tr, Div, eanI
Div, aenI	sanF, Inf	Σ, Tr, Div, aenI
Div, Inf	sanF	Σ, Tr, Div, Inf

Theorem 33. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves the sets in the first column of Table 2 but not the sets in the second column, and $\circ\circlearrowleft\tau \cong \tau\circlearrowright\circ \xrightarrow{\tau} \circ$, then it is the equivalence induced by the sets in the third column.

Proof. Let [r1] to [r6] refer to the rows in the table.

Lemmas 2 [r3], 12 [r1], 13 [r1,2], 14 [r4,5,6], and 15 [r4,5,6] imply that if "$\cong$" preserves the sets in the first column, then "$\cong$" also preserves the additional sets in the third column.

To prove the first claim that "$\cong$" can be no other equivalence, let f be the f of Lemma 17 and h be the h of Lemma 32. We have $L \cong f(L) \cong h(f(L))$,

$$\begin{array}{rclcll}
Sf(h(f(L))) & = & Tr(f(L)) \times 2^{\Sigma(f(L))} & = & (anT(L) \cup extT(L)) \times 2^{\Sigma(L)} & , \\
Div(h(f(L))) & = & Div(f(L)) & = & extT(L) & \text{, and} \\
Inf(h(f(L))) & = & Inf(f(L)) & = & anI(L) \cup extI(L) & .
\end{array}$$

Because $extT(L)$ and $extI(L)$ are functions of $\Sigma(L)$ and $minD(L)$, Lemma 3 applies and gives the claim.

The remaining five claims that "$\cong$" can be no other equivalence are proven in a similar way using the f from Lemmas 23(b) [r2], 23(a) [r3], 26 [r4], and 28 [r5], and the function $f(L) = L$ [r6]. In all cases $Sf(h(f(L))) = Tr(f(L)) \times 2^{\Sigma(f(L))} = Tr(L) \times 2^{\Sigma(L)}$. Depending on the case, $Div(h(f(L)))$ is $Tr(L) \cap extT(L)$ [r2,3] or $Div(L)$ [r4,5,6], and $Inf(h(f(L)))$ is $anI(L)$ [r2], $eanI(L)$ [r4], $aenI(L)$ [r5], or $Inf(L)$ [r3,6]. $\square$

The *weakest livelock-preserving congruence* is the weakest congruence that guarantees for every L and L' that if $Div(L) = \emptyset \neq Div(L')$, then $L \not\cong L'$. In [13] it was proven that the weakest livelock-preserving congruence with respect to $L \setminus A$ and $L \,||\, L'$ is the equivalence induced by Σ, anT, $minD$, and anI. Only equivalences that preserve Σ were considered. In the present publication, the apparently weaker starting point $\circ \not\cong \tau\circlearrowright\circ \xrightarrow{\tau} \circ$ was used and the same result was obtained as Theorem 16 and Lemmas 12 and 13. When taking the preservation of Σ as an assumption, their proofs only use $L \setminus A$ and $L \,||\, L'$. The equivalence induced by Σ, Tr, Div, and $eanI$ is the weakest congruence with respect to $L \setminus A$ and $L \,||\, L'$ that preserves divergence traces [13]. This result corresponds to Lemmas 14 and 15.

After adding $sanF$, there is no unique next set of stable failures, but two. So we need two different functions that throw out some information on stable failures while preserving the congruence.

The function h_1 in the next lemma throws away all information on stable failures at and after minimal divergence traces. To facilitate the use of the lemma in two different situations, it has two alternative assumptions on Div.

Lemma 34. Assume that "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves Σ but not snF, and $\circ\circlearrowleft\tau \cong \tau\circlearrowright\circ \xrightarrow{\tau} \circ$. For every LTS L such that $Div(L) = Tr(L) \cap extT(L)$ there is an LTS $h_1(L)$ such that $h_1(L) \cong L$, $Sf(h_1(L)) = anF(L) \cup (\,(Tr(L) \cap extT(L)) \times 2^{\Sigma(L)}\,)$, $Div(h_1(L)) = Div(L)$, and $Inf(h_1(L)) = Inf(L)$. If "$\cong$" preserves Div, then the assumption $Div(L) = Tr(L) \cap extT(L)$ is not needed.

Proof. Let ν be from Lemma 30. If $Div(L) = Tr(L) \cap extT(L)$, then ν qualifies as the h_1.

The case remains where "$\cong$" preserves Div. Let $M_1 \cong M_2$, $(\sigma, A) \in snF(M_1) \setminus snF(M_2)$, $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$, $b_1 \cdots b_n = \sigma^{[1]}$, $\{a_1, \ldots, a_m\} = A^{[1]}$, $c = 1^{[0]}$, and $\Sigma_L = \Sigma(L)$. Let $T_{\sigma,A}$ be like in Fig. 12 with the alphabet $\{c\} \cup \Sigma_M^{[1]} \cup \Sigma_L^{[2]}$. Let $g(L)$ be $\lceil \nu(L) \rceil^{[2]}$ with a c-transition added from each divergent state to itself. When $i \in \{1,2\}$, let

$$f_i(L) \;=\; \lfloor\, (g(L) \,||\, T_{\sigma,A} \,||\, c.\lceil M_i \rceil^{[1]}) \setminus (\{c\} \cup \Sigma_M^{[1]})\, \rfloor_{[2]} \;.$$

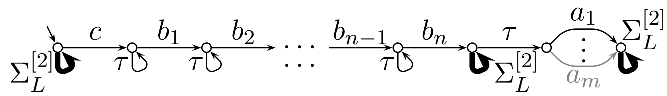


Figure 12: An LTS for detecting a strongly nondivergent failure. The thick arrows denote that there is a transition for each $a \in \Sigma_L^{[2]}$.

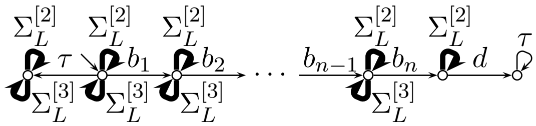
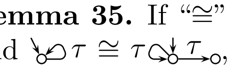


Figure 13: An LTS for detecting an always nondivergent failure.

By construction, $f_i(L)$ can do everything that $\nu(L)$ can do, but it can also hiddenly execute c from any divergent state. After executing c, $f_i(L)$ tries to hiddenly execute $\sigma^{[1]}$. If that fails, then $f_i(L)$ is trapped in a divergence. If that succeeds, then $g(L)$ can continue but $T_{\sigma,A}$ is in an unstable state and M_i is at an end state of σ. We have $\mathit{Inf}(f_i(L)) = \mathit{Inf}(L)$. By the definition of snF, M_1 does not diverge when $T_{\sigma,A}$ is in any of its last three states, but if $T_{\sigma,A}$ continues, then $T_{\sigma,A} \,||\, c.\lceil M_1 \rceil^{[1]}$ may deadlock. Thanks to the use of ν, also $g(L)$ may enter a stable state, resulting in a total deadlock. So $f_1(L)$ behaves otherwise like L, but has also the stable failures $(\mathit{Tr}(L) \cap \mathit{extT}(L)) \times 2^{\Sigma(L)}$.

Because "$\cong$" preserves Div, M_2 cannot cause a divergence when $T_{\sigma,A}$ is in any of its last three states. It cannot cause a deadlock either, because $(\sigma, A) \notin \mathit{snF}(M_2)$. So $f_2(L) \doteq \nu(L)$. In conclusion, $L \cong \nu(L) \doteq f_2(L) \cong f_1(L)$, and $f_1(L)$ qualifies as the h_1. □

The function h_2 in the next lemma throws away all information on stable failures whose trace is or whose refused action would complete a divergence trace. Its construction requires that no state is the end state of both a divergence trace and a nondivergent trace. To cope with this problem, we use the function Una defined in the previous section.

Lemma 35. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves minD but not anF, and , then for every LTS L there is an LTS $h_2(L)$ such that $h_2(L) \cong L$, $\mathit{Sf}(h_2(L)) = (\mathit{Div}(L) \times 2^{\Sigma(L)}) \cup \{(\sigma, A_1 \cup A_2) \mid (\sigma, A_1) \in \mathit{snF}(L) \wedge \forall a \in A_2 : \sigma a \in \mathit{Div}(L)\}$, $\mathit{Div}(h_2(L)) = \mathit{Div}(L)$, and $\mathit{Inf}(h_2(L)) = \mathit{Inf}(L)$.

Proof. By Theorem 1, "$\cong$" preserves Σ. Let $M_1 \cong M_2$, $(\sigma, A) \in \mathit{anF}(M_1) \setminus \mathit{anF}(M_2)$, $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$, $b_1 \cdots b_n = \sigma^{[1]}$, and $d = 1^{[0]}$. Let L be any LTS and $\Sigma_L = \Sigma(L)$. Let T_σ^d be the LTS with the alphabet $\{d\} \cup \Sigma_M^{[1]} \cup \Sigma_L^{[2]} \cup \Sigma_L^{[3]}$ whose graph is in Fig. 13. When $i \in \{1, 2\}$, let

$$M_i' \;=\; (\,(T_\sigma^d \,||\, M_i \Phi^{[1],d}) \setminus \Sigma_M^{[1]}\,)\Phi_d^{[3]} \;,$$

where $\Phi_d^{[3]}$ renames d to each $x \in \Sigma_L^{[3]}$, and $\Phi^{[1],d}$ renames each $x \in \Sigma_M$ to $x^{[1]}$ and each $x \in A$ also to d. If $A = \emptyset$, we let $M_i \Phi^{[1],d} = \lceil M_i \rceil^{[1]} \,||\, \mathsf{Stop}\{d\}$, so that d is not accidentally left out from the alphabet $\Sigma_M^{[1]} \cup \{d\}$. Clearly M_i refuses A if and only if $M_i \Phi^{[1],d}$ refuses d. Let $\Xi = \Sigma_L^{[2]} \cup \Sigma_L^{[3]}$. We have $\Sigma(M_i') = \Xi$.

Clearly $\mathit{Inf}(M_1') = \mathit{Inf}(M_2') = \Xi^\omega$. Because $(\sigma, A) \in \mathit{anF}(M_1)$ and "$\cong$" preserves minD, M_i' cannot diverge before executing d. The leftmost state of T_σ^d is stable, ensuring $(\sigma, \emptyset) \in \mathit{Sf}(M_i')$ for every $\sigma \in \Xi^*$. No other states of T_σ^d can add to $\mathit{Sf}(M_i')$, except perhaps the start state of the d-transition. Because M_2 cannot execute σ or refuse A after it, $T^d_\sigma \mid\mid M_2\Phi^{[1],d}$ cannot refuse d after $b_1 \cdots b_n$. So $\mathit{Sf}(M'_2) = \{(\sigma, \emptyset) \mid \sigma \in \Xi^*\}$. Since M_1 can, $\mathit{Sf}(M'_1) = \{(\sigma, B) \mid \sigma \in \Xi^* \wedge B \subseteq \Sigma^{[3]}_L\}$.

Let $g(L)$ be $\mathsf{Una}(L)$ with each visible label x replaced by $x^{[3]}$ if the transition ends in a potentially divergent state, and $x^{[2]}$ otherwise. Let $\Phi_{[2,3]}$ rename each $x^{[2]}$ and $x^{[3]}$ to x. Consider $f_i(L) = (g(L) \mid\mid M'_i)\Phi_{[2,3]}$. When M'_i diverges, d has just been executed. Thus $g(L)$ has just completed a divergence trace and M'_i blocks the visible transitions. So $\mathit{Div}(f_i(L)) = \mathit{Div}(L)$. M'_2 does not affect the behaviour of $g(L)$ in any other way, so $f_2(L) \doteq L$. On the other hand, M'_1 can block all actions that would complete a nonempty divergence trace of L. So

$$\mathit{Sf}(f_1(L)) \;=\; \{(\sigma, B_1 \cup B_2) \mid (\sigma, B_1) \in \mathit{Sf}(L) \wedge \forall a \in B_2 : \sigma a \in \mathit{Div}(L)\} \;.$$

It implies

$$\begin{aligned} \mathit{nF}(f_1(L)) &= \{(\sigma, B_1 \cup B_2) \mid (\sigma, B_1) \in \mathit{nF}(L) \wedge \forall a \in B_2 : \sigma a \in \mathit{Div}(L)\} \\ &= \{(\sigma, A_1 \cup A_2) \mid (\sigma, A_1) \in \mathit{snF}(L) \wedge \forall a \in A_2 : \sigma a \in \mathit{Div}(L)\} \;, \end{aligned}$$

where the last equality is obtained by letting $A_2 = \{a \in B_1 \cup B_2 \mid \sigma a \in \mathit{Div}(L)\}$ and $A_1 = (B_1 \cup B_2) \setminus A_2$.

As a consequence, $\nu(f_1(L))$ qualifies as the $h_2(L)$, where ν is from Lemma 30. □

The following theorem lists the next six congruences and points direction to the next nine.

Table 3: The congruences of Theorem 36

preserves	does not preserve	induced by
sanF, minD	Tr, anF	Σ, sanF, minD, anI
Tr, sanF, minD	anF, Div, Inf	Σ, Tr, sanF, minD, anI
sanF, minD, Inf	anF, Div	Σ, Tr, sanF, minD, Inf
sanF, Div	anF, snF, aenI	Σ, Tr, sanF, Div, eanI
sanF, Div, aenI	anF, snF, Inf	Σ, Tr, sanF, Div, aenI
sanF, Div, Inf	anF, snF	Σ, Tr, sanF, Div, Inf

Theorem 36. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves the sets in the first column of Table 3 but not the sets in the second column, and $\circ\!\circlearrowleft\tau \cong \tau\circlearrowright\!\circ\xrightarrow{\tau}\circ$, then "$\cong$" is the equivalence induced by the sets in the third column.

Proof. Let [r1] to [r6] refer to the rows in the table.

Lemmas 2 [r3], 13 [r1,2], 14 [r4,5,6], and 15 [r4,5,6] imply that if "$\cong$" preserves the sets in the first column, then it preserves also the additional sets in the third column.

To prove the claims that "$\cong$" can be no other equivalence, let f' be the f in Lemma 17 [r1], 23(b) [r2], 23(a) [r3], 26 [r4], or 28 [r5], or the function $f'(L) = L$ [r6]. So $\mathit{Div}(f'(L))$ is either $\mathit{extT}(L)$ [r1], $\mathit{Tr}(L) \cap \mathit{extT}(L)$ [r2,3], or $\mathit{Div}(L)$ [r4,5,6]; and $\mathit{Inf}(f'(L))$ is either $\mathit{anI}(L) \cup \mathit{extI}(L)$ [r1], $\mathit{anI}(L)$ [r2], $\mathit{eanI}(L)$ [r4], $\mathit{aenI}(L)$ [r5], or $\mathit{Inf}(L)$ [r3,6]. Furthermore, $\mathit{Sf}(f'(L)) = \mathit{Sf}(L)$ and $f'(L) \cong L$. Because "$\cong$" preserves minD or Div, we have $\mathit{minD}(f'(L)) = \mathit{minD}(L)$ and $\mathit{extT}(f'(L)) = \mathit{extT}(L)$. Excluding [r1], we also have $\mathit{Tr}(f'(L)) = \mathit{Tr}(L)$.

Let h_1 and h_2 be like in Lemmas 34 and 35, and let $f(L) = h_2(h_1(f'(L)))$. The validity of some assumptions of Lemma 34 is not immediately obvious, so let us check them. By

Lemma 14, *Div* is not preserved on [r1]. By Lemma 31, *snF* is not preserved on [r1,2,3]. It is explicitly given in the table that *snF* is not preserved on [r4,5,6]. On [r4,5,6], *Div* is preserved. We show next that [r1,2,3] satisfy $Div(f'(L)) = Tr(f'(L)) \cap extT(f'(L))$. By Lemma 23, [r2,3] have $Div(f'(L)) = Tr(L) \cap extT(L) = Tr(f'(L)) \cap extT(f'(L))$. By Lemma 17, [r1] has $Div(f'(L)) = extT(L) = extT(f'(L)) = Tr(f'(L)) \cap extT(f'(L))$, because $Div(f'(L)) \subseteq Tr(f'(L))$ by the definition of *Div*. So Lemma 34 can be used.

We have $L \cong f'(L) \cong h_1(f'(L)) \cong f(L)$, $Div(f(L)) = Div(f'(L))$, and $Inf(f(L)) = Inf(f'(L))$. All assumptions of Lemma 3 can now be checked except the $Sf(f(L))$ assumption. To facilitate checking it, too, we show next that $Sf(f(L)) = F(L)$, where

$$
\begin{aligned}
F(L) \;\; = \;\; & \big(\big(Tr(f'(L)) \cap extT(f'(L)) \big) \times 2^{\Sigma(L)} \big) \cup \\
& \{ (\sigma, A_1 \cup A_2) \mid (\sigma, A_1) \in sanF(f'(L)) \wedge \forall a \in A_2 : \sigma a \in minD(f'(L)) \} \; .
\end{aligned}
$$

Let $\sigma \in Tr(f'(L))$, $A \subseteq \Sigma(L)$, $A_2 = \{a \in A \mid \sigma a \in Div(f'(L))\}$, and $A_1 = A \setminus A_2$.

Assume first that $\sigma \in extT(f'(L))$. Then clearly $(\sigma, A) \in F(L)$. By Lemma 34, $(\sigma, A) \in Sf(h_1(f'(L)))$. If $\sigma \in Div(h_1(f'(L)))$, then the first part and otherwise the second part of the expression for $Sf(h_2(\ldots))$ in Lemma 35 yields $(\sigma, A) \in Sf(f(L))$.

In the remaining case $\sigma \notin extT(f'(L))$. That implies $\sigma \in anT(f'(L))$. Then $\sigma a \in minD(f'(L))$ if and only if $\sigma a \in Div(f'(L))$ if and only if $\sigma a \in Div(h_1(f'(L)))$. Furthermore, $(\sigma, A) \in F(L)$ if and only if $(\sigma, A_1) \in sanF(f'(L))$ if and only if $(\sigma, A_1) \in sanF(h_1(f'(L)))$ if and only if $(\sigma, A_1) \in snF(h_1(f'(L)))$ if and only if $(\sigma, A) \in Sf(f(L))$.

We have shown $Sf(f(L)) = F(L)$.

Because f' preserves *Sf* and *minD*, we have $sanF(f'(L)) = sanF(L)$. On [r1],

$$
Tr(f'(L)) \;=\; Div(f'(L)) \cup Sf^{\,Tr}(f'(L)) \;=\; extT(L) \cup Sf^{\,Tr}(L) \;=\; extT(L) \cup sanF^{\,Tr}(L) \; ,
$$

because if $\sigma \notin extT(L)$ and $(\sigma, \emptyset) \in Sf(L)$, then $(\sigma, \emptyset) \in sanF(L)$. In the remaining cases "$\cong$" preserves *Tr*, so $Tr(f'(L)) = Tr(L)$. Thus Lemma 3 applies in all cases. □

The equivalence induced by Σ, *sanF*, *minD*, and *anI* is the weakest "any-lock"-preserving congruence (that is, the weakest congruence that distinguishes LTSs that can stop executing visible actions from those that cannot) with respect to $L \setminus A$ and $L \,||\, L'$, as was proven in [12].

Six more congruences follow.

Table 4: The congruences of Theorem 37

preserves	does not preserve	induced by
anF, *minD*	*Tr*	Σ, *anF*, *minD*, *anI*
Tr, *anF*, *minD*	*Div*, *Inf*	Σ, *Tr*, *anF*, *minD*, *anI*
anF, *minD*, *Inf*	*Div*	Σ, *Tr*, *anF*, *minD*, *Inf*
anF, *Div*	*snF*, *aenI*	Σ, *Tr*, *anF*, *Div*, *eanI*
anF, *Div*, *aenI*	*snF*, *Inf*	Σ, *Tr*, *anF*, *Div*, *aenI*
anF, *Div*, *Inf*	*snF*	Σ, *Tr*, *anF*, *Div*, *Inf*

Theorem 37. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves the sets in the first column of Table 4 but not the sets in the second column, and $\tau \cong \tau \; \tau$, then "$\cong$" is the equivalence induced by the sets in the third column.

Proof. The proof is like the proof of Theorem 36 with the following differences. Now h_2 is not used, so $f(L) = h_1(f'(L))$. By the definition of h_1,

$$\mathit{Sf}(f(L)) \;=\; \mathit{anF}(f'(L)) \cup (\, (\mathit{Tr}(f'(L)) \cap \mathit{extT}(f'(L))) \times 2^{\Sigma(L)}\,)\ .$$

We have $\mathit{anF}(f'(L)) = \mathit{anF}(L)$. On [r1], $\mathit{Tr}(f'(L)) = \mathit{extT}(L) \cup \mathit{sanF}^{\mathit{Tr}}(L) = \mathit{extT}(L) \cup \mathit{anF}^{\mathit{Tr}}(L)$. □

The congruence induced by Σ, *anF*, *minD*, and *anI* is the same as the well-known failures-divergences equivalence in the CSP theory [16]. It is more often defined by requiring that Σ, *CFail*, and *CDiv* are preserved, where (in our terminology) $\mathit{CDiv}(L) = \mathit{extT}(L)$ and $\mathit{CFail}(L) = \mathit{Sf}(L) \cup (\,\mathit{CDiv}(L) \times 2^{\Sigma(L)})$. That *anI* is preserved is not required, because the LTSs are assumed to be finitely branching, that is, for every s, the set $\{s' \mid \exists a : (s, a, s') \in \Delta\}$ is finite. It makes *anI* a function of *anT*. Often other parallel composition operators than the one defined in this publication are used, making it unnecessary to talk about Σ.

In CSP theory, the congruence was defined using a fixed-point method that gives a meaning to recursively defined process expressions without appealing to LTSs. A natural consequence of this method is that the resulting congruence preserves no information beyond minimal divergence traces. With it, each divergence is equivalent to $\mathsf{RDL}(\Sigma_L)$ in Fig. 1. This phenomenon is called *catastrophic divergence* and $\mathsf{RDL}(\Sigma_L)$ is called *chaos*. The phenomenon is harmful in many applications. This motivated the development and name of CFFD-equivalence, that is, chaos-free failures divergences equivalence. Recently, a complicated fixed-point definition for the equivalence induced by Σ, *Tr*, *Div*, and *eanI* has been found [15]. To this, *Sf* can be added.

7.3. **(Strongly) nondivergent failures.** We still have to consider the congruences that preserve *snF* or more and satisfy $\tau \cong \tau\ \tau$. There are three groups of them. Again, each group corresponds to Section 6. However, because of Lemma 31, each group only contains congruences that preserve *Div*, so it only contains three congruences.

Table 5: The congruences of Theorem 38

preserves	does not preserve	induced by
snF	*anF*, *aenI*	Σ, *snF*, *Div*, *eanI*
snF, *aenI*	*anF*, *Inf*	Σ, *snF*, *Div*, *aenI*
snF, *Inf*	*anF*	Σ, *snF*, *Div*, *Inf*

Theorem 38. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves the sets in the first column of Table 5 but not the sets in the second column, and $\tau \cong \tau\ \tau$, then "$\cong$" is the equivalence induced by the sets in the third column.

Proof. Lemmas 31 and 15 imply that "$\cong$" preserves *Div*, Σ, and *eanI*.

To prove the claims that "$\cong$" can be no other equivalence, let f' be the f in Lemma 26 or 28, or the function $f'(L) = L$. Let h_2 be like in Lemma 35, and let $f(L) = h_2(f'(L))$. We have $L \cong f(L)$, $\mathit{Div}(f(L)) = \mathit{Div}(L)$, $\mathit{Inf}(f(L)) = \mathit{Inf}(f'(L))$, and

$$\mathit{Sf}(f(L)) = (\mathit{Div}(L) \times 2^{\Sigma(L)}) \cup \{(\sigma, A_1 \cup A_2) \mid (\sigma, A_1) \in \mathit{snF}(L) \wedge \forall a \in A_2 : \sigma a \in \mathit{Div}(L)\}\ .$$

Furthermore, $\mathit{Inf}(f'(L))$ is either $\mathit{eanI}(L)$, $\mathit{aenI}(L)$, or $\mathit{Inf}(L)$. So Lemma 3 applies. □

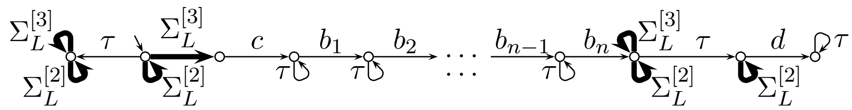


Figure 14: An LTS for detecting a nondivergent failure.

Lemma 39. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves *Div* but not *nF*, and $\circlearrowleft\tau \cong \tau\circlearrowright\xrightarrow{\tau}\circ$, then for every LTS L there is an LTS $h(L)$ such that $h(L) \cong L$, $\mathit{Div}(h(L)) = \mathit{Div}(L)$, $\mathit{Inf}(h(L)) = \mathit{Inf}(L)$, and

$$\begin{aligned} \mathit{Sf}(h(L)) \;\; &= \;\; \mathit{anF}(L) \cup (\mathit{Div}(L) \times 2^{\Sigma(L)}) \cup \\ &\qquad \{(\sigma, A_1 \cup A_2) \mid (\sigma, A_1) \in \mathit{snF}(L) \wedge \sigma \in \mathit{extT}(L) \wedge \forall a \in A_2 : \sigma a \in \mathit{Div}(L)\} \ . \end{aligned}$$

Proof. Theorem 1 implies that "$\cong$" preserves Σ. Let $M_1 \cong M_2$, $(\sigma, A) \in \mathit{nF}(M_1) \setminus \mathit{nF}(M_2)$, $\Sigma_M = \Sigma(M_1) = \Sigma(M_2)$, $b_1 \cdots b_n = \sigma^{[1]}$, $c = 1^{[0]}$, and $d = 2^{[0]}$. Let L be any LTS and $\Sigma_L = \Sigma(L)$. Let T_σ^d be the LTS whose alphabet is $\{c, d\} \cup \Sigma_M^{[1]} \cup \Sigma_L^{[2]} \cup \Sigma_L^{[3]}$ and whose graph is in Fig. 14. When $i \in \{1, 2\}$, let

$$M_i' \;=\; (\ (T_\sigma^d \,||\, c.(M_i \Phi^{[1],d})\ ) \setminus (\{c\} \cup \Sigma_M^{[1]})\ )\Phi_d^{[3]} \ ,$$

where $\Phi^{[1],d}$ renames each $x \in \Sigma_M$ to $x^{[1]}$ and each $x \in A$ also to d, and $\Phi_d^{[3]}$ renames d to each $x \in \Sigma_L^{[3]}$. We use the same trick as in the proof of Lemma 35 to ensure that $\Sigma(M_i \Phi^{[1],d}) = \Sigma_M^{[1]} \cup \{d\}$ even if $A = \emptyset$. Let $\Xi = \Sigma_L^{[2]} \cup \Sigma_L^{[3]}$. We have $\Sigma(M_i') = \Xi$.

Clearly $\mathit{Inf}(M_1') = \mathit{Inf}(M_2') = \Xi^\omega$. Because $(\sigma, A) \in \mathit{nF}(M_1)$ and "$\cong$" preserves *Div*, $\sigma \notin \mathit{Div}(M_1) = \mathit{Div}(M_2)$. Therefore, $\mathit{Div}(M_1') = \mathit{Div}(M_2') \subseteq \{\sigma a \mid \sigma \in \Xi^* \wedge a \in \Sigma_L^{[3]}\}$. The leftmost state of T_σ^d is stable, ensuring $(\sigma, \emptyset) \in \mathit{Sf}(M_i')$ for every $\sigma \in \Xi^*$. No other states of T_σ^d can affect $\mathit{Sf}(M_i')$, except perhaps the start state of the d-transition. Because M_2 cannot execute σ or refuse A after it, $T_\sigma^d \,||\, c.(M_2 \Phi^{[1],d})$ cannot refuse d after $b_1 \cdots b_n$. Therefore, $\mathit{Sf}(M_2') = \{(\sigma, \emptyset) \mid \sigma \in \Xi^*\}$. However, M_1 can, so we have $\mathit{Sf}(M_1') = \{(\sigma, \emptyset) \mid \sigma \in \Xi^*\} \cup \{(\sigma a \rho, B) \mid \sigma\rho \in \Xi^* \wedge a \in \Sigma_L^{[3]} \wedge B \subseteq \Sigma_L^{[3]}\}$.

Let $g(L)$ be $\mathsf{Una}(L)$ with each visible label x replaced by $x^{[3]}$ if the transition ends in a potentially divergent state of $g(L)$, and $x^{[2]}$ otherwise. Let $\Phi_{[2,3]}$ rename each $x^{[2]}$ and $x^{[3]}$ to x. Consider $f_i(L) = (g(L) \,||\, M_i')\Phi_{[2,3]}$. When M_i' diverges, also $g(L)$ completes a divergence trace and M_i' blocks the visible transitions. M_2' does not affect the behaviour of $g(L)$ in any other way, so $f_2(L) \doteq L$. On the other hand, M_1' can block all actions that would complete a nonminimal divergence trace.

Let ν be like in Lemma 30. Clearly $\nu(f_1(L)) \cong L$, $\mathit{Div}(\nu(f_1(L))) = \mathit{Div}(L)$, and $\mathit{Inf}(\nu(f_1(L))) = \mathit{Inf}(L)$. By analysing in turn the stable failures whose trace is always-nondivergent, divergent, or neither of them, we see that

$$\begin{aligned} \mathit{Sf}(\nu(f_1(L))) \;\; &= \;\; \mathit{anF}(L) \cup (\mathit{Div}(L) \times 2^{\Sigma(L)}) \cup \\ &\qquad \{(\sigma, A_1 \cup A_2) \mid (\sigma, A_1) \in \mathit{snF}(L) \wedge \sigma \in \mathit{extT}(L) \wedge \forall a \in A_2 : \sigma a \in \mathit{Div}(L)\}. \end{aligned}$$

Thus $\nu(f_1(L))$ qualifies as the $h(L)$ of the claim. $\square$

Table 6: The congruences of Theorem 40

preserves	does not preserve	induced by
anF, *snF*	*nF*, *aenI*	Σ, *anF*, *snF*, *Div*, *eanI*
anF, *snF*, *aenI*	*nF*, *Inf*	Σ, *anF*, *snF*, *Div*, *aenI*
anF, *snF*, *Inf*	*nF*	Σ, *anF*, *snF*, *Div*, *Inf*

Theorem 40. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves the sets in the first column of Table 6 but not the sets in the second column, and $\searrow\!\circ\!\circlearrowleft\tau \cong \tau\circlearrowright\!\downarrow\!\circ\xrightarrow{\tau}\circ$, then "$\cong$" is the equivalence induced by the sets in the third column.

Proof. The proof is like the proof of Theorem 38, but using the h of Lemma 39 instead of the h_2 of Lemma 35. $\square$

Table 7: The congruences of Theorem 41

preserves	does not preserve	induced by
nF	*Sf*, *aenI*	Σ, *nF*, *Div*, *eanI*
nF, *aenI*	*Sf*, *Inf*	Σ, *nF*, *Div*, *aenI*
nF, *Inf*	*Sf*	Σ, *nF*, *Div*, *Inf*

Theorem 41. If "$\cong$" is a congruence, "$\doteq$" implies "$\cong$", "$\cong$" preserves the sets in the first column of Table 7 but not the sets in the second column, and $\searrow\!\circ\!\circlearrowleft\tau \cong \tau\circlearrowright\!\downarrow\!\circ\xrightarrow{\tau}\circ$, then "$\cong$" is the equivalence induced by the sets in the third column.

Proof. The proof is like the proof of Theorem 38, but using the ν of Lemma 30 instead of the h_2 of Lemma 35. $\square$

The equivalence induced by Σ, *nF*, *Div*, and *eanI* is the weakest congruence that preserves all traces that can lead to an "any-lock" (that is, deadlock or livelock) with respect to $L \setminus A$ and $L \mid\mid L'$, as was proven in [12]. The same (pre)congruence is the weakest that preserves so-called conditional liveness properties [4]. The equivalence induced by Σ, *nF*, *Div*, and *Inf* is called *nondivergent failures divergences equivalence* or *NDFD-equivalence*. In [8] it was proven that it is the weakest congruence that preserves all properties that can be formulated in the stuttering-insensitive linear temporal logic of [10]. A variant of this result, where the logic is connected to LTSs in a more intuitive way, was presented in [18].

A comparison of the "induced by" and "does not preserve" colums of Table 2 to 7 reveals that all possibilities with $\searrow\!\circ \not\cong \searrow\!\circ\!\circlearrowleft\tau \cong \tau\circlearrowright\!\downarrow\!\circ\xrightarrow{\tau}\circ$ have been investigated.

8. Conclusion

Fig. 15 shows the relations between the abstract linear-time congruences discussed in this publication as a Hasse diagram. There are altogether 40 of them. If the set of considered operators is $a.L$, $L \setminus A$, $L\Phi$, and $L \mid\mid L'$, then for any stuttering-insensitive linear-time property, its optimal congruence is among those in the figure.

For instance, what is the weakest linear-time congruence that distinguishes $\searrow\!\circ\xrightarrow{a}\circ$ from $\circ\xleftarrow{\tau}\searrow\!\circ\xrightarrow{a}\circ$? Clearly the equivalence induced by Σ, *Tr*, *Div*, and *Inf* does not separate them. This also rules out the nine equivalences that are connected downstream to it in the figure. On the other hand, the equivalence induced by Σ and *Sf* separates them, and so does the

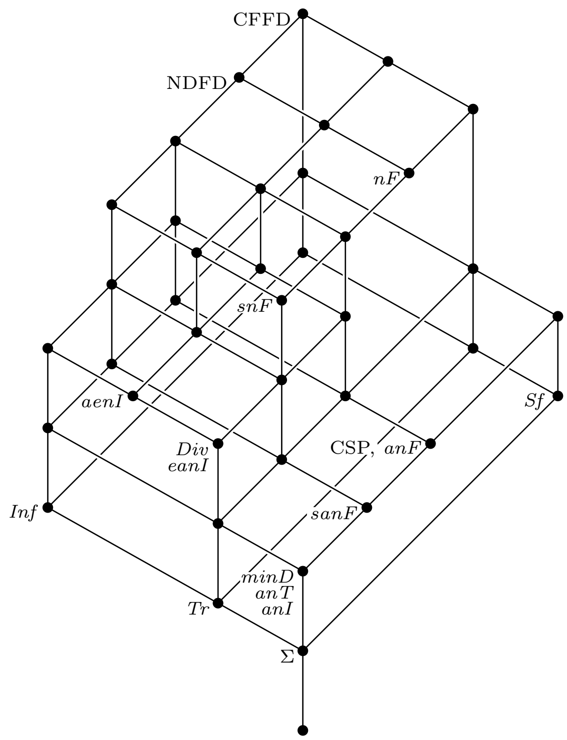


Figure 15: All abstract linear-time congruences with respect to $a.L$, $L \setminus A$, $L\Phi$, and $L \,||\, L'$. Names in *italics* indicate the new preserved set(s). Other names are the names of the congruences. There is a path from "$\cong_1$" down to "$\cong_2$" if and only if "$\cong_1$" implies "$\cong_2$".

equivalence induced by Σ, *sanF*, *minD*, and *anI*. So there is no unique weakest linear-time congruence, but two. It is worth mentioning that outside linear-time, also observation equivalence [11] separates them, although it is not strictly stronger than the two linear-time congruences mentioned above.

With a smaller set of operators, there may be more congruences. With a bigger set, there may be fewer. However, it may also be that "$\doteq$" is not a congruence with respect to the bigger set. Then it is necessary to strengthen "$\doteq$". This makes room for more congruences. This happens if the "choice" operator of CCS is employed. Then one must add one bit to the semantics that tells if the initial state is stable [22, 23]. This splits some congruences in the figure to two, one with and another without the initial stability bit.

If the LTSs are finite, then the distinction between *Tr*, *aenI*, and *Inf* disappears, because then the infinite traces are determined by the traces, as shown by (3.3). Then some congruences merge, leaving 20 distinct congruences.

Acknowledgement

I thank Rob van Glabbeek, Bill Roscoe, and the anonymous reviewers for helpful comments. In particular, the anonymous reviewers found problems and suggested fixes in Lemma 34 and its proof.

References

[1] Bolognesi, T., Brinksma, E.: Introduction to the ISO Specification Language LOTOS. *Computer Networks and ISDN Systems*, vol. 14, pp. 25–59 (1987)

[2] De Nicola, R., Vaandrager, F.: Three Logics for Branching Bisimulation. *Journal of the ACM* 42(2), 458–487 (1995)

[3] Gazda, M., Fokkink, W.: Congruence from the Operator's Point of View: Compositionality Requirements on Process Semantics. In: Aceto, L., Sobocinski, P. (eds.) *Proc. Seventh Workshop on Structural Operational Semantics*. Electronic Proceedings in Theoretical Computer Science 32, 15–25 (2010)

[4] van Glabbeek, R.: The Coarsest Precongruences Respecting Safety and Liveness Properties. In: Calude, C.S., Sassone, V. (eds.) *Proc. Theoretical Computer Science — 6th IFIP TC 1/WG 2.2 Int. Conf., TCS 2010*. IFIP AICT 323, Springer, 32–52 (2010)

[5] van Glabbeek, R.: The Linear Time — Branching Time Spectrum II: The Semantics of Sequential Systems with Silent Moves. In: Best, E. (ed.) *Proc. CONCUR '93, Fourth International Conference on Concurrency Theory*, Lecture Notes in Computer Science, vol. 715, 66–81 (1993)

[6] Graf, S., Steffen, B., Lüttgen, G.: Compositional Minimisation of Finite State Systems Using Interface Specifications. *Formal Aspects of Computing* 8(5), 607–616 (1996)

[7] Hoare, C.A.R.: *Communicating Sequential Processes*. Prentice-Hall, Englewood Cliffs, NJ, 256 p. (1985)

[8] Kaivola, R., Valmari, A.: The Weakest Compositional Semantic Equivalence Preserving Nexttime-less Linear Temporal Logic. In: Cleaveland, R. (ed.) *Proc. CONCUR '92, Third International Conference on Concurrency Theory*, Lecture Notes in Computer Science, vol. 630, 207–221 (1992)

[9] Madelaine, E., Vergamini, D.: AUTO: A Verification Tool for Distributed Systems Using Reduction of Finite Automata Networks. In: Vuong, S.T. (ed.) *Formal Description Techniques II (FORTE '89)*, North-Holland, 61–66 (1990)

[10] Manna, Z., Pnueli, A.: *The Temporal Logic of Reactive and Concurrent Systems, Volume I: Specification*. Springer, Heidelberg, 427 p. (1992)

[11] Milner, R.: *Communication and Concurrency*. Prentice-Hall, Englewood Cliffs, NJ, 260 p. (1989)

[12] Puhakka, A.: Weakest Congruence Results Concerning "Any-Lock". In: Kobayashi, N., Pierce, B.C. (eds.) *TACS 2001, Fourth International Symposium on Theoretical Aspects of Computer Software*, Lecture Notes in Computer Science, vol. 2215, 400–419 (2001)

[13] Puhakka, A., Valmari, A.: Weakest-Congruence Results for Livelock-Preserving Equivalences. In: Baeten, J.C.M., Mauw, S. (eds.) *CONCUR '99, 10th International Conference on Concurrency Theory*, Lecture Notes in Computer Science, vol. 1664, 510–524 (1999)

[14] Rensink, A., Vogler, W.: Fair Testing. *Information and Computation* 205(2), 125–198 (2007)

[15] Roscoe, A.W.: Seeing Beyond Divergence. In: Abdallah, A.E., Jones, C.B., Sanders, J.W. (eds.) *Communicating Sequential Processes. The First 25 Years*, Lecture Notes in Computer Science, vol. 3525, 15–35 (2005)

[16] Roscoe, A.W.: *Understanding Concurrent Systems*. Springer, Heidelberg, 533 p. (2010)

[17] Valmari, A.: The Weakest Deadlock-Preserving Congruence. *Information Processing Letters* 53(6), 341–346 (1995)

[18] Valmari, A.: A Chaos-Free Failures Divergences Semantics with Applications to Verification. *Millennial Perspectives in Computer Science, Proceedings of the 1999 Oxford–Microsoft Symposium in Honour of sir Tony Hoare*, Palgrave, 365–382 (2000)

[19] Valmari, A.: Composition and Abstraction. In: Cassez, F., Jard, C., Rozoy, B., Ryan, M.D. (eds.) *Modeling and Verification of Parallel Processes*, LNCS Tutorials, Lecture Notes in Computer Science, vol. 2067, 58–98 (2001)

[20] Valmari, A.: All Linear-Time Congruences for Finite LTSs and Familiar Operators. In: Brandt, J., Heljanko, K. (eds.) *Proc. Application of Concurrency to System Design, 12th Int. Conf.*, IEEE, 12–21 (2012)

[21] Valmari, A.: All Linear-Time Congruences for Familiar Operators Part 2: Infinite LTSs. In: Koutny, M., Ulidowski, I. (eds.) *Proc. CONCUR 2012, 23rd International Conference on Concurrency Theory*, Lecture Notes in Computer Science, vol. 7454, 162–176 (2012)

[22] Valmari, A., Tienari, M.: An Improved Failures Equivalence for Finite-State Systems with a Reduction Algorithm. In: Jonsson, B., Parrow, J., Pehrson, B. (eds.) *Proc. Protocol Specification, Testing and Verification XI*, North-Holland, 3–18 (1991)

[23] Valmari, A., Tienari, M.: Compositional Failure-Based Semantic Models for Basic LOTOS. *Formal Aspects of Computing* 7(4), 440–468 (1995)