The following full text is a preprint version which may differ from the publisher's version.

For additional information about this publication click this link.
http://hdl.handle.net/2066/32308

Please be advised that this information was generated on 2018-06-15 and may be subject to change.
The Coarsest Congruence for Timed Automata with Deadlines Contained in Bisimulation

Pedro R. D’Argenio\textsuperscript{1**} and Biniam Gebremichael\textsuperscript{2}

\textsuperscript{1} CONICET – FaMAF, Universidad Nacional de Córdoba.
Ciudad Universitaria, 5000 Córdoba, Argentina.
dargenio AT famaf.unc.edu.ar

\textsuperscript{2} Institute for Computing and Information Sciences
Radboud University Nijmegen
P.O. Box 9010, 6500 GL Nijmegen, The Netherlands
B.Gebremichael AT cs.ru.nl

Abstract. Delaying the synchronisation of actions may reveal some hidden behaviour that would not happen if the synchronisation would meet the specified deadlines. This precise phenomenon makes bisimulation fail to be a congruence for the parallel composition of timed automata with deadlines, a variant of timed automata where time progress is controlled by deadlines imposed on each transition. This problem has been known and unsolved for several years. In this paper we give a characterisation of the coarsest congruence that is included in the bisimulation relation. In addition, a symbolic characterisation of such relation is provided and shown to be decidable. We also discuss the pitfalls of existing parallel compositions in this setting and argue that our definition is both reasonable and sufficiently expressive as to consider the modelling of both soft and hard real-time constraints.

1 Introduction

Design and specification languages allow to model systems in a modular manner by linking small modules or components using the language operations—such as the sequential composition or the parallel composition—in order to build larger modules. Hence a desirable requirement is that the language is compositional with respect to its semantics. By compositional we mean that components can be replaced by behaviorally equivalent components without changing the properties of the larger model in which they are embedded. The preservation of such properties can be guaranteed by means of semantic equivalences or preorders. For example branching bisimulation preserves CTL* \cite{11}, language inclusion preserves LTL \cite{20} and, in particular, timed bisimulation preserves (timed) properties expressed in logics such as TCTL \cite{25}. Hence, compositionality amounts to requiring that relations like these are congruences (or precongruences) for the different operations of the language.

\textsuperscript{*} This work was supported by the European Community Project IST-2001-35304 AMETIST, http://ametist.cs.ssu.twente.nl.

\textsuperscript{**} Part time researcher at Formal Methods and Tools Group, Dep. of Comp. Sci. University of Twente. Supported by the NWO Vernieuwingsimpuls project “Verification of performance and dependability” and the ANPCyT project PICT 11-11738 “Teoría y Herramientas para la Construcción de Software Crítico”.
Real time systems can be modeled using timed automata [2, 17] which have become popular as modelling language for several model checkers [3, 9, 10] because of its simplicity and tractability. Timed automata are automata with the additional ingredients of clocks. Clocks are variables that increase at the same rate in order to register time progress. Transitions of timed automata are labelled with constraints on clocks, called guards, that indicate when such transition may take place. Usually timed automata are used to model real-time systems with hard constraints. In this cases, timed automata are equipped with an invariant, which is a constraint on clocks that limits time progress in each control state [17]: the system is obliged to leave such state before invalidating the invariant.

Timed automata with deadlines (TAD for short) [24, 7, 5, 6] were introduced to simplify the compositionality problem in timed systems, allowing also the modelling of soft real time systems. At the same time, the TAD model ensures, under reasonable assumption, what is called time reactivity in [6] and time lock freedom in [8], that is, whenever time progress stops there exists at least one transition enabled. This model is nowadays embedded in modelling languages such as IF [10] and MoDeST [15], and urgent transitions in Uppaal [4] can be seen as a particular instance of TAD transitions.

TAD do not have invariants. Instead, a TAD transition has associated a second clock constraint, called deadline, that indicates in which moment such transition must be taken. As a consequence, a deadline is required to hold only if the corresponding guard holds ensuring the transition can be taken after the deadline is reached. In this sense, the deadline impose an urgency constraint.

Contrary to the traditional timed automata setting, bisimulation in the TAD model is not preserved by parallel composition [6]. This is illustrated in the following example. $T_1$ in Fig. 1.(a) depicts a TAD in which circles represent control state and arrows are control transitions. In particular the small incoming arrow identifies the initial state. $T_1$ performs first an action $b$ at any moment and sets clock $x$ to 0. As time progresses, the value of $x$ increases and when it takes value 2 action $a$ becomes enabled. This is controlled by guard $\gamma : x \geq 2$. At any point after $x$ takes value 2, this transition may take place, but as time continue to progress and $x$ takes value 3, the deadline $\delta : x \geq 3$ obliges the execution of the transition. Notice that $T_2$ shows a similar behaviour since action $c$ cannot be executed: the deadline of $a$ obliges its execution before the

![Fig. 1. TAD and compositionality](image-url)
guard of $c$ becomes enabled. In fact, $T_1$ and $T_2$ are timed bisimilar in the sense of [6].

Suppose now that $T_1$ is composed in parallel with the automaton $\text{stop}$ requiring synchronisation on action $a$. ($\text{stop}$ is the automaton with a single location and no transition; hence, it does not do anything but idling.) This blocks the execution of action $a$ in $T_1$. The resulting automaton $T_1 \ ||_a \text{stop}$ is depicted in Fig. 1.(b). Similarly, the composition of $T_2$ with $\text{stop}$ in $T_2 \ ||_a \text{stop}$ also blocks the execution of $a$, but in this case time progresses beyond 3 time units allowing the execution of $c$ after 4 time units (see Fig. 1.(b)). As a consequence $T_1 \ ||_a \text{stop}$ and $T_2 \ ||_a \text{stop}$ are not bisimilar.

To the best of our knowledge there does not exist a characterisation of a congruence for parallel composition on TADs. The only exception is what is called strong congruence in [6], which is the usual bisimulation applied directly on TADs. This relation is, however, far too strong as it requires the syntactic equality of guards, deadlines, and clock resets.

In this paper we present a congruence relation for parallel composition and prove that it is the coarsest congruence included in the bisimulation relation. This new relation, which we call $\nabla$-bisimulation (read “drop-bisimulation”), is in fact the usual bisimulation on an extended semantics of TAD. Such semantics allows for time progressing beyond deadlines but carefully accounting the actions whose deadline have been overruled. We also give a symbolic characterisation of $\nabla$-bisimulation, that is, a relation defined directly on TADs. As a corollary of this characterisation we obtain that $\nabla$-bisimulation is decidable. Another particular contribution of this paper is that the proof of congruence is entirely carried out at symbolic level (i.e., without resorting to the underlying transition system in which $\nabla$-bisimulation is defined). We finally discuss different kind of parallel compositions on TADs (mostly defined already in the literature) reporting which of them preserves $\nabla$-bisimulation and which not and why.

Related Work. The failure of bisimulation to be a congruence becomes apparent when soft deadlines are considered, that is actions that may be urgent in isolation are required to wait if they are intended for synchronisation i.e. synchronising actions need to be patient. This same problem has appeared in the context of stochastic process algebra where synchronisation is required to be patient (see e.g. [19, 18, 14]). The problem becomes evident (in a similar manner as above) if bisimulation is considered for the underlying probabilistic transition system rather than for the finer symbolic model [14]. The problem of compositionality also showed up in other process algebras for performance behaviour [13].

Recently, [16] defined a variant of TADs where actions are distinguished between input and output following the model of [23] and for which bisimulation is a congruence for the parallel composition. This is possible due to input enabledness and to the fact that only output actions are allowed to be urgent (i.e. to have deadline.) Therefore there is no need to wait for synchronisation as it is always possible. Though the restrictions imposed by [16] makes this new
model much simpler and tractable, using it to describe soft real-time systems may result in complex models.

In addition to the solution for the compositionality problem, we also give a symbolic characterisation of the congruence. Our work is based on the result of Lin & Yi [21] who gave a symbolic characterisation of the bisimulation for timed automata. In turn, their result is based on Čeráns’ who determined that bisimulation for timed automata is decidable [12]. We use also this result to show the decidability of the $\nabla$-bisimulation.

Paper Outline. The paper is organized as follows. Section 2 gives the preliminaries recalling timed automata with deadlines, its semantics in terms of transition systems, the definition of bisimulation, and particularly, the definition of parallel composition. In Section 3 we discuss the pitfalls of the composition and progressively construct the semantics that leads to the definition of $\nabla$-bisimulation. The symbolic characterisation is provided in Section 4 and shown to be the coarsest congruence in Section 5. We conclude in Section 6 discussing decidability of $\nabla$-bisimulation and the different kind of synchronisation in parallel composition.

2 Preliminaries

In this section we recall the definition of timed automata with deadlines, their semantics and composition.

Timed Automata with Deadlines. A clock is a non-negative real-valued variable, which can be reset to zero at the occurrence of an event, and between two resets, its derivative with respect to time is equal to 1. We denote $\mathcal{C} = \{x_1, \ldots, x_N\}$ to be a finite set of clocks. A clock constraint $\mathcal{F}(\mathcal{C})$ is a conjunction of formula(s) of atomic constraints in the form of $x_i \leq n$ or $x_i - x_j \geq m$, where $x_i$ and $x_j$ are clocks in $\mathcal{C}$, $\leq, \geq, <, >, $ and $n, m$ are natural numbers. The constraints $\top$ and $\bot$ are used to denote, respectively, the atomic constraints which are constantly true and false. We will assume that there is a global finite set of actions $\mathcal{A}$ for all timed automata with deadlines.

Definition 1. A timed automaton with deadlines [6] (TAD for short) is a structure $T = (\mathcal{L}, I^0, \mathcal{C}, \rightarrow)$ where

- $\mathcal{L}$ is a finite set of locations,
- $I^0 \subseteq \mathcal{L}$ is the set of initial locations,
- $\mathcal{C}$ is a finite set of clocks,
- $\rightarrow \subseteq \mathcal{L} \times (\mathcal{A} \times \mathcal{F}(\mathcal{C}) \times \mathcal{F}(\mathcal{C}) \times 2^\mathcal{C}) \times \mathcal{L}$, is a finite set of edges

If $(s, a, \gamma, \delta, \mathbf{x}, s') \in \rightarrow$ we write $s \xrightarrow{a, \gamma, \delta, \mathbf{x}} s'$ and require that $\delta \Rightarrow \gamma$ holds, moreover we assume $\delta$ is left-closed (left-closure is formally defined in Def. 2).

The notion $s \xrightarrow{a, \gamma, \delta, \mathbf{x}} s'$ represents an edge from location $s$ to $s'$ that executes action $a$ whenever guard $\gamma$ becomes true. In addition, deadline predicate $\delta$ impose an urgency condition: the transition cannot be delayed whenever $\delta$ is satisfied. When executing the transition, clocks in $\mathbf{x}$ are set to 0.
Parallel composition of TADs. Parallel composition allows the independent execution of the activity of the component automata except if they are intended to synchronise. We assume CSP synchronisation in which actions with equal name synchronise if and only if they belong to a set of synchronizing actions \( B \subseteq A \). Since enabling of actions is determined by guards, we define the guard on the synchronised transition to be the conjunction of the guards on the synchronising transitions. Therefore synchronisation take place only if both partners are able to execute the same synchronising action. (Other compositions of guards are discussed in Sec. 6). Similarly, the deadlines of the synchronising transitions should affect the deadline of the synchronisation. In this case, we do not fix any particular operation. Instead, we assume a given operator \( \otimes \) that applied to guards and deadlines of the synchronising transitions returns the deadline of the synchronisation. We require that \( \otimes \) satisfies the following:

1. \((\delta_1, \gamma_1) \otimes (\delta_2, \gamma_2) \Rightarrow (\gamma_1 \land \gamma_2)\) whenever \( \delta_1 \Rightarrow \gamma_1 \) and \( \delta_2 \Rightarrow \gamma_2 \)
2. \( \otimes \) preserves left-closure, that is \((\delta_1, \gamma_1) \otimes (\delta_2, \gamma_2)\) is left closed whenever \( \delta_1 \) and \( \delta_2 \) are left closed.
3. \( \otimes \) distributes with respect to \( \lor \) in all its arguments, that is
   \[
   (\lor_i (\delta^i_1, \gamma^i_1) \otimes (\delta^i_2, \gamma^i_2)) \Leftrightarrow (\lor_i (\delta^i_1, i \gamma^i_1) \otimes (\lor_i (\delta^i_2, \gamma^i_2))
   \]
4. There exists a constraint \( C_B \) such that \((0_B, \text{tt})\) act as a neutral element for \( \otimes \) in the following sense: \((\delta_1, \gamma_1) \otimes (0_B, \text{tt})) \Rightarrow \delta_1 \)

\((\delta_1, \gamma_1) \otimes (\delta_2, \gamma_2)\) has to imply the guard \( \gamma_1 \land \gamma_2 \) of the resulting transition in order to preserve this property on the composed TAD. This is required in 1. Similarly the left-closure property of the deadline is required in 2, inorder to preserve this property. The distributivity of 3 is needed to prove congruence (see proof of Theorem 2). As we will see in the next section, time passage in a location is limited by the complement of the disjunction of the outgoing deadlines. Therefore condition 3 states compositionality for \( \otimes \), allowing to represent the deadline of a synchronised action in terms of the deadlines and guards of the component automata. Constraint 4 is only necessary to show that our definition is the coarsest congruence included in the bisimulation (see Lemma 6).

For operators not meeting this condition there may exist coarser congruences than ours that are also bisimulation. Constraint 4 guarantees a way to test the validity of the original deadline in a component’s transition by means of a synchronisation. In section 6 we discuss different implementations of \( \otimes \).

Let \( T_i = (L_i, \ell^0_i, C_i, \rightarrow_i) \), such that \( C_i \cap C_2 = \emptyset \) for \( i \in \{1, 2\} \), and let \( B \subseteq A \) be a set of synchronising actions, and \( \otimes \) be an operation for synchronising deadlines. The parallel composition \( T_1 \| B T_2 \) is defined by the TAD \((L_1 \times L_2, \ell^0_1 \times \ell^0_2, C_1 \cup C_2, \rightarrow)\) where \( \rightarrow \) is defined as the smallest relation satisfying:

\[
\begin{align*}
&\frac{s_1 \xrightarrow{a, \gamma, \delta, x} s'_1 \quad a \notin B}{(s_1, s_2) \xrightarrow{\alpha, \gamma, \delta, x} (s'_1, s_2)} & \frac{s_2 \xrightarrow{a, \gamma, \delta, x} s'_2 \quad a \notin B}{(s_1, s_2) \xrightarrow{\alpha, \gamma, \delta, x} (s'_1, s_2)} \\
&\frac{s_1 \xrightarrow{a_1 \gamma_1, \delta_1, x_1} s'_1 \quad s_2 \xrightarrow{a_2 \gamma_2, \delta_2, x_2} s'_2 \quad a \in B}{(s_1, s_2) \xrightarrow{a \gamma_1 \land \gamma_2, (\delta_1 \gamma_1) \otimes (\delta_2 \gamma_2), x_1 \cup x_2} (s'_1, s'_2)}
\end{align*}
\]
The rules are fairly standard. Notice, in particular, that the last rule only allows to synchronise guards when both of them are valid. This is a significant restriction w.r.t. [6]. We later argue that our restriction is reasonable and discuss the feasibility of compositions not consider here. From now on, we will omit the subscript on edges as they will be clear from the context.

**Transition Systems and Bisimulation.** A transition system (TS for short) is a structure \( TS = (S, s^0, \Sigma, \rightarrow) \) where \( S \) is an infinite set of states, \( s^0 \) is the set of initial states, \( \Sigma \) is a set of labels, and \( \rightarrow \subseteq (S \times \Sigma \times S) \) is a set of transitions. Since we use TSs to model timed systems, we consider two kind of labels: those representing the execution of discrete actions and those representing the passage of time. Then \( \Sigma = A \cup \mathbb{R}_{\geq 0} \). We also expect that these timed TSs satisfy Wang’s properties [26] (though this is not relevant for the present result).

A bisimulation [22] is a symmetric relation \( R \in S \times S \) such that for all \( \ell \in \Sigma \), whenever \( (p, q) \in R \) and \( p \xrightarrow{\ell} p' \) then \( q \xrightarrow{\ell} q' \) and \( (p', q') \in R \) for some \( q' \). We write \( p \xrightarrow{\ell} q \) if \( (p, q) \in R \) for some bisimulation relation \( R \) on TS. Given two TSs \( T_1 \) and \( T_2 \) with set of initial states \( s_1^0 \) and \( s_2^0 \), respectively, we say that they are bisimilar (notation \( T_1 \sim T_2 \)) if there is a bisimulation \( R \) in the disjoint union of \( T_1 \sqcup T_2 \) such that \( s_j^0 \subseteq R(s_i^0) \) for \( \{i, j\} = \{1, 2\} \), i.e. every initial state of \( T_1 \) is related to some initial state of \( T_2 \) and vice-versa.

**Semantics of TADs.** In the following we recall the semantics of TADs in terms of TSs. A state of the timed system is divided in two parts, one indicating the current control location in the TAD, and the other the current time values. This last part is represented by means of a clock valuation which is a function \( \rho : \mathcal{C} \rightarrow \mathbb{R}_{\geq 0} \) mapping to each clock the time elapsed since the last time it was reset to 0. Given a clock valuation \( \rho \) and \( d \in \mathbb{R}_{\geq 0} \) the function \( \rho + d \) denotes the valuation such that for each clock \( x \in \mathcal{C} \), \( (\rho + d)(x) = \rho(x) + d \). The function \( \rho(x:0) \) denotes the valuation such that for each clock \( x \in x \cap \mathcal{C} \), \( \rho(x:0)(x) = 0 \), otherwise \( \rho(x:0)(x) = \rho(x) \).

We first define what it means for a constraint to be left-closed, followed by the semantics of TADs.

**Definition 2.** A constraint \( \phi \) is called left closed if and only if for all valuations \( \rho \),

\[
\rho \models \lnot \phi \Rightarrow \exists \varepsilon > 0 : \forall \varepsilon' \leq \varepsilon : \rho + \varepsilon' \models \lnot \phi
\]

**Definition 3.** Let \( T = (\mathcal{L}, s^0, \mathcal{C}, \rightarrow) \) be a TAD. Its semantics is given by \( TS(T) = (\mathcal{L} \times (\mathcal{C} \rightarrow \mathbb{R}_{\geq 0}), \mathcal{I} \times (\mathcal{C} ightarrow 0), A \cup \mathbb{R}_{\geq 0}, \rightarrow) \), where \( \rightarrow \) is the smallest relation satisfying these two rules:

**A1:** discrete transition

\[
s \xrightarrow{a;\gamma;\delta;x} s' \text{ and } \rho \models \gamma \text{ implies } sp \xrightarrow{a} s'\rho(x:0)
\]

**A2:** delay transition

\[
\forall d' < d : \rho + d' \models tpc(s) \text{ implies } sp \xrightarrow{d} s(\rho + d)
\]

where \( tpc(s) \) is the timed progress condition in location \( s \) defined by

\[
tpc(s) = \lnot \bigvee \{ \delta \mid s \xrightarrow{a;\gamma;\delta;x} s' \text{ for some } a, \gamma, x, s' \}
\]
Rule A1 states that an edge $s \xrightarrow{a;\gamma;\delta;\kappa} s'$ defines a discrete transition in current location $s$ whenever the guard holds in current valuation $\rho$. After the transition is taken clocks in $x$ are set to 0 in the new valuation. According to A2, time can progress in $s$ only when $tpc(s)$ is true, that is as long as no deadline of an edge leaving $s$ becomes true. Notice that $tpc(s)$ is required to hold for all $d' < d$ but not for $d$ itself. Therefore it is indistinguishable whether $tpc(s)$ holds in the limit or not. For instance, if $\rho(x) = 0$ both $x < 3$ and $x \leq 3$ hold in all $\rho + d'$ with $d' < 3$. Thus our assumption that deadline has to be specified as left-closed predicate is just a preference not a limitation.

As a consequence of Def. 3 the notion of bisimulation extends to TADs straightforwardly: two TADs $T_1$ and $T_2$ are bisimilar (notation $T_1 \sim T_2$) if $TS(T_1) \sim TS(T_2)$.

**Example 1.** Consider automata $T_1$ and $T_2$ of Fig. 1. Using Def. 3 it is routine to check that relation

$$R = \{(s_0\{x:=d\}, t_0\{x:=d\}) \mid 0 \leq d\}
\cup \{(s_1\{x:=d\}, t_1\{x:=d\}) \mid 0 \leq d \leq 3\}
\cup \{(s_2\{x:=d\}, t_2\{x:=d\}) \mid 2 \leq d\}$$

is a bisimulation witnessing $T_1 \sim T_2$. Besides, if $stop = (\{r\}, \{r\}, \emptyset, \emptyset)$, then $T_2 \parallel a\odot stop$ can execute the trace $b5c$ through the execution

$$(t_0, r)\{x:=0\} \xrightarrow{b} (t_1, r)\{x:=0\} \xrightarrow{5} (t_1, r)\{x:=5\} \xrightarrow{c} (t_3, r)\{x:=5\}$$

which is not possible in $(s_0, r)\{x:=0\}$. Consequently, $T_1 \parallel a\odot stop \neq T_2 \parallel a\odot stop$.

### 3 Towards a Congruence Relation

In the following we discuss different proposals for congruence until finding a satisfactory definition. All proposals are bisimulation relations on different modifications of the transition system underlying the TAD.

The running example (Fig. 1) suggest that action $c$ could have been distinguished if time would be allowed to elapse beyond the deadline. Therefore, a first naive proposal would be to let time progress beyond the time progress condition but this would not be compatible with the bisimulation since TADs with different deadlines but equal guards may become equated. So, a modification of this semantics could be to consider separately a potential time progress by adding a new kind of transition: $s\rho \xrightarrow{[d]} s(\rho + d)$ for all $d \geq 0$. Though clearly stronger than bisimulation —notice that it would distinguish $T_1$ and $T_2$ in Fig. 1— it fails to be a congruence. The example that shows it is given in Fig. 2(a). The relation would equate $T_3$ and $T_4$, but not their compositions $T_3 \parallel a\odot T'$ and $T_4 \parallel a\odot T'$ with $B = \{a, b, c\}$. Notice that after realisation of action $a$, $T_3 \parallel a\odot T'$ lets (non-potential) time progress beyond 2 time units while this is not possible in $T_4 \parallel a\odot T'$ due to the deadline in $b$.

As consequence of this, we may think to consider different potential time progress transition for each edge in the TAD, but this turns to be too strong.
Fig. 2. (Counter)examples for congruence

(a) T_3

(b) T_4

(c) T_5

( composing automata )

Let A_V = \{ \nabla_D \mid D \subseteq A \}. To keep track of which deadlines have to be disregarded, states also need to bookkeep the current set of actions whose deadlines were dropped. The extended semantics of T = (L, \emptyset, C, \rightarrow) is then given by the TS (L \times 2^A \times (C \mapsto \mathbb{R}_\geq 0), \emptyset \times \{ \emptyset \} \times (C \mapsto 0), A \cup A_V \cup \mathbb{R}_\geq 0, \rightarrow), where \rightarrow is the smallest relation satisfying:

A1: discrete transition
s \xrightarrow{a, \gamma, \delta, x} s' and \rho \models \gamma \implies (s, D)\rho \xrightarrow{a} (s', \emptyset)\rho(x:=0)

A2: delay transition
\forall d' < d : \rho + d' \models -dl(s, A - D) \implies (s, D)\rho \xrightarrow{d} (s, D)\rho + d

A3: drop transition
(s, D)\rho \xrightarrow{\nabla_E} (s, D \cup E)\rho

where dl(s, A) is the deadline collected by actions in A \subseteq A in location s and is defined by

\[ dl(s, A) = \bigvee \{ \delta \mid s \xrightarrow{a, \gamma, \delta, x} s' \text{ and } a \in A \text{ for some } a, \gamma, x, s' \} \]
Bisimulation in this new semantics distinguishes automata in Fig. 1(a) and Fig. 2(a), and equates those in Fig. 2(b). Regarding to the new predicate $dl(s, A)$ notice that for any location $s$, $tpc(s) = \neg dl(s, A)$.

Notice that once a deadline is dropped, it cannot be observed anymore. Example in Fig. 2(c) shows that this semantics does not yet yields a congruence. According to this semantics $T_7$ and $T_8$ are equated. However, under the assumption that deadlines and guards of synchronising transitions are arranged ample in Fig. 2(c) shows that this semantics does not yet yields a congruence.

is the smallest relation satisfying $\text{Bisimulation in this new semantics distinguishes automata in Fig. 1(a) and notice that for any location $s$, $tpc(s) = \neg dl(s, A)$.}$

Notice that the undrop action can be performed at any moment. Notice also $(s_1, \emptyset) \rho \xrightarrow{\Delta} (s_2, \emptyset) \rho$.

$\text{A4: undrop transition } (s, D) \rho \xrightarrow{\Delta} (s', \emptyset) \rho$.

Notice that the undrop action can be performed at any moment. Notice also that the execution sequence $a \Downarrow_{\{b\}} 2 \Delta 1$ is possible in $T_8$ but not in $T_7$. Hence, a bisimulation in this setting distinguishes $T_7$ from $T_8$. We define such a relation as follows.

**Definition 4**. The extended semantics of $T = (L, \emptyset, C, \rightarrow)$ is given by $TS^e(T) = (L \times 2^A \times (C \rightarrow \mathbb{R}_{\geq 0}), \emptyset \times \{\emptyset\} \times (C \rightarrow 0), A \cup A^e \cup \{\emptyset\} \cup \mathbb{R}_{\geq 0}, \rightarrow)$, where $\rightarrow$ is the smallest relation satisfying $A1_\emptyset$, $A2_\emptyset$, and $A3$ above plus $A4$.

$\text{A4: undrop transition } (s, D) \rho \xrightarrow{\Delta} (s', \emptyset) \rho$.

Notice that the undrop action can be performed at any moment. Notice also that the execution sequence $a \Downarrow_{\{b\}} 2 \Delta 1$ is possible in $T_8$ but not in $T_7$. Hence, a bisimulation in this setting distinguishes $T_7$ from $T_8$. We define such a relation as follows.

**Definition 5** ($\nabla$-bisimulation). We say that automata $T_1$ and $T_2$ are $\nabla$-bisimilar, notation $T_1 \sim_\nabla T_2$, if $TS^e(T_1) \sim TS^e(T_2)$. We also say that locations $s$ and $t$ are $\nabla$-bisimilar in some valuation $\rho$, notation $s \rho \sim_\nabla t \rho$, if $(s, \emptyset) \rho \sim_\nabla (t, \emptyset) \rho$.

**Proposition 1**. For any $T_1$ and $T_2$, if $T_1 \sim_\nabla T_2$ then $T_1 \sim T_2$.

**Proof.** It is routine to check that if $R$ is a bisimulation that witness $T_1 \sim_\nabla T_2$, then $\{(s_1, \rho_1, s_2, \rho_2) \mid ((s_1, \emptyset) \rho_1, (s_2, \emptyset) \rho_2) \in R\}$ is a bisimulation that witness $T_1 \sim T_2$. \qed

We conclude this section by proving two basic properties of $\nabla$-bisimulation. These properties (lemmas) will be later used to prove Theorem 1, that relates $\sim_\nabla$ to a symbolic bisimulation.

Notice that the ability of dropping all the deadlines, letting time pass, and then undropping the deadlines, ensures that if two locations are $\nabla$-bisimilar at a certain moment, no matter how long the activity is blocked, this two locations will still be $\nabla$-bisimilar. This is stated in the following lemma.

**Lemma 1**. If $(t, \emptyset) \rho \sim (u, \emptyset) \rho$ then $(t, \emptyset)(\rho + d) \sim (u, \emptyset)(\rho + d)$, for all $d \geq 0$. 

9
We postpone the proof that we first give a symbolic characterisation of region. A clock region or in TADs which does resort to the underlying transition system and equates exactly the same automata as constraints of the form, 4 Symbolic Characterisation of

Proof. By A3 in Def. 4, (t, ϕ) A (t, A)A which implies (u, ϕ) A (u, A)A and (t, A)A (u, A)A (briefly a bisimulation.)

Since d(t, ϕ) = \sqrt{ϕ} = fe, \rho + d \models -d(t, A - A), for all d \geq 0. Hence, by A2ϕ, (t, A)A d (t, A)(ρ + d) for any d \geq 0. This implies that (u, A)A d (u, A)(ρ + d) and (t, A)(ρ + d) (u, A)(ρ + d) for all d \geq 0.

Finally, since (t, A)(ρ + d) A (t, ϕ)(ρ + d) by A4, (u, A)(ρ + d) A (u, ϕ)(ρ + d) and (t, ϕ)(ρ + d) (u, ϕ)(ρ + d) for all d \geq 0.

If two locations are ϕ-bisimilar at some given valuation ρ then both satisfy the deadline associated to some action in valuation ρ, or none of them does. This is easy to check by dropping all the deadlines except those associated to the action of interest. This is formally stated in the next lemma.

Lemma 2. If (t, ϕ)A (u, ϕ)A then ρ \models dl(t, D) ⇔ dl(u, D), for any D ⊆ A.

Proof. Let (t, ϕ)A (u, ϕ)A. By A3 (t, ϕ)A ϕ A (t, A - D)A which implies that (u, ϕ)A ϕ A (u, A - D)A and (t, A - D)A (u, A - D)A.

We show that (t, A - D)A (u, A - D)A implies ρ \models dl(t, D) ⇒ dl(u, D) which, by symmetry of \models, suffices to show that ρ \models dl(t, D) ⇔ dl(u, D).

\[ \rho \models -dl(u, D) \]
\[ ⇒ \exists d > 0 : \forall d' : 0 \leq d' < d : \rho + d' \models -dl(t, D) \quad \text{(by Def. 2)} \]
\[ ⇒ (u, A - D)A d \rho \quad \text{(by A2ϕ)} \]
\[ ⇒ (t, A - D)A d \rho \quad \text{(since (t, A - D)A (u, A - D)A)} \]
\[ ⇒ \exists d > 0 : \forall d' : 0 \leq d' < d : \rho + d' \models -dl(t, D) \quad \text{(by A2ϕ)} \]
\[ ρ \models -dl(t, D) \quad \text{(in particular, for } d' = 0) \]

4 Symbolic Characterisation of ϕ-bisimulation

We postpone the proof that ϕ-bisimulation is a congruence until Section 5 and give first a symbolic characterisation of ϕ-bisimulation. That is, we give a relation directly in TADs which does resort to the underlying transition system and equates exactly the same automata as ϕ-bisimulation does.

The symbolic bisimulation we propose works in a similar fashion to that of [21]. The construction of such relation is based on zone and region manipulation. A clock region or region for short, is a consistent conjunction of atomic constraints of the form,

\[ ψ ≡ \bigwedge_{x \in C} ψ_x \land \bigwedge_{(x, y) \in C, x ≠ y} ψ_{(x, y)} \]

where

- each ψ_x is either x = n, m < x < m + 1 or x > N, and
- each ψ_{(x, y)} is either x - y = n, m < x - y < m + 1 or x - y > N.
with \( n, m, N \) non-negative integers such that \( 0 \leq n \leq N \), and \( 0 \leq m < N \).

Regions can be expressed by constraints as we defined above, and any constraint can be expressed as a disjunction of regions.

Similar to the clock resetting (\( \rho\{x := 0\} \)) and time successor \( \rho + d \) of the clock valuation defined earlier, we define below their symbolic counterpart.

**Reset:** For a constraint \( \phi \) and a set of clocks \( x \), the reset \( \phi|_x \) is a predicate such that for all \( \rho \),
\[
\rho \models \phi|_x \iff \rho = \rho'|_x = 0 \text{ and } \rho' \models \phi \text{ for some } \rho'
\]

**Time successor:** For a constraint \( \phi \), the time successor \( \phi \uparrow \) is a predicate such that for all \( \rho \),
\[
\rho \models \phi \uparrow \iff \rho = \rho' + d \text{ and } \rho' \models \phi \text{ for some } \rho' \text{ and } d \geq 0
\]

A constraint \( \phi \) is \( \uparrow \)-closed if and only if \( \phi \uparrow \leftrightarrow \phi \) is valid (i.e. a tautology).

The operations above distribute on disjunction and are expressible in terms of constraints (see e.g. \([27,21]\)). The following facts can be derived from the definitions or have already appear elsewhere \([27,21]\).

**Fact 1.** 1. Let \( \psi \) and \( \phi \) be regions. Let \( \rho \) and \( \rho' \) be valuations s.t. \( \rho \models \psi \) and \( \rho' \models \psi \). If \( \rho + d \models \phi \) for some \( d \geq 0 \), there exists \( d' \geq 0 \) such that \( \rho' + d' \models \phi \).
2. If \( \phi \) is a region then, for any constraint \( \psi \), either \( \phi \Rightarrow \psi \) is valid or \( \phi \land \psi \) is a contradiction.
3. If \( \phi \) is a region, so does \( \phi|_x \).
4. \( \rho \models \phi \) implies \( \rho \models \phi \uparrow \).
5. \( \phi \uparrow \) is \( \uparrow \)-closed.
6. If \( \phi \) is \( \uparrow \)-closed then \( \rho \models \phi \) implies \( \rho + d \models \phi \) for all \( d \in \mathbb{R}_{\geq 0} \).
7. If \( \phi_1 \) and \( \phi_2 \) are \( \uparrow \)-closed (resp. left-closed), so are \( \phi_1 \land \phi_2 \) and \( \phi_1 \lor \phi_2 \).

Given a constraint \( \phi \), a \( \phi \)-partition \([21]\) is a finite set of constraints \( \Phi \) if \( \bigvee \Phi \leftrightarrow \phi \) and for any two distinct \( \psi, \psi' \in \Phi \), \( \psi \) and \( \psi' \) are disjoint (i.e. \( \psi \land \psi' \) is a contradiction). A \( \phi \)-partition \( \Phi \) is called finer than another \( \phi \)-partition \( \Psi \) if \( \Phi \) can be obtained from \( \Psi \) by decomposing some of its elements. \( \mathcal{RC}(\phi) \) denotes the set of all regions that constitute \( \phi \). Notice that \( \phi \leftrightarrow \bigvee \mathcal{RC}(\phi) \) and that \( \mathcal{RC}(\phi) \) is the finest of all \( \phi \)-partitions.

**Lemma 3.** Let \( \psi \) be a region and \( \rho \) be such that \( \rho \models \psi \). For all \( \phi \in \mathcal{RC}(\psi \uparrow) \) exists \( d \geq 0 \) such that \( \rho + d \models \phi \).

**Proof.** Let \( \rho'' \models \phi \), then \( \rho'' \models \psi \uparrow \). By the definition of \( \uparrow \), exists \( \rho' \) and \( d' \geq 0 \) such that \( \rho' + d' = \rho'' \) and \( \rho' \models \psi \). Since \( \rho \models \psi \) too, and \( \psi \) and \( \phi \) are regions, by Fact 1.1, exists \( d \geq 0 \) such that \( \rho + d \models \phi \).

The definition of symbolic bisimulation we propose is based on Lin & Yi’s definition \([21]\), which in turns is based on Čeräns’ result \([12]\). A symbolic bisimulation is a relation containing tuples \((s, t, \phi)\) meaning that locations \( s \) and \( t \) are related in any valuation that satisfies constraint \( \phi \). Here \( \phi \) is a constraint over the disjoint union of the set of clocks of the two automata. In this way, the relation ensures that clocks in both automata progress at the same rate. In turn, this guarantees that the related locations can idle the same time until some given deadline becomes true.
Definition 6 (Symbolic Bisimulation). Let $T_1$ and $T_2$ be two TADs with disjoint set of clocks $C_1$ and $C_2$ and disjoint set of locations $L_1$ and $L_2$ respectively. A relation $S \subseteq (L_1 \times L_2 \cup L_2 \times L_1) \times F(C_1 \cup C_2)$ (where $F(C)$ denotes the set of all constraints with clocks in $C$) is a symbolic bisimulation if for all $(t, u, \phi) \in S$,

1. $(u, t, \phi) \in S$,
2. $\phi$ is $\upharpoonright$-closed,
3. whenever $t \xrightarrow{a, \gamma, \delta, x} t'$, there is a $(\phi \land \gamma)$-partition $\Phi$ such that for each $\phi' \in \Phi$, $u \xrightarrow{a, \gamma', \delta', y} u'$, $\phi' \Rightarrow \gamma'$ and $(t', u', \phi' \downharpoonright_{xy}) \in S$, for some $\gamma'$, $\delta'$, $y$ and $u'$; and
4. $\phi \Rightarrow (dl(t, A) \Leftrightarrow dl(u, A))$ is valid for all $A \subseteq A$.

We write $t \sim^\phi u$ if $(t, u, \phi) \in S$ for some symbolic bisimulation $S$. We also write $T_1 \sim^\phi T_2$ if for every initial location $t$ of $T_1$ there is an initial location $u$ in $T_2$ such that $t \sim^\phi u$, and the same with the roles of $T_1$ and $T_2$ exchanged.

Property 1 states the symmetric characteristics of a bisimulation. The requirement that $\phi$ is $\upharpoonright$-closed (property 2) ensures that location $t$ and $u$ show an equivalent behaviour any time in the future which is necessary if deadlines are dropped. Property 3 ensures the transfer properties of discrete transitions. This is similar to [21] except that there is no invariant to consider. Finally, property 4 states that any possible combination of deadlines should match under the assumption that $\phi$ holds. This ensures that the time elapsed until a deadline associated to a given action is the same in both locations. Notice that property 4 is equivalent to requiring that $\phi \Rightarrow (dl(t, \{a\}) \Leftrightarrow dl(u, \{a\}))$ for all $a \in A$. This makes evident that deadlines may be “changed” from one edge to another as long as both edges are labelled with the same action (see Fig. 2(b)). Moreover property 4 is comparable to the property of invariants in [21]. Like in [21], the use of partitioning allows that one edge is matched by several edges as is the case in Fig. 3 where both $T_9 \sim^N T_{10}$ and $T_9 \sim^{x=y} T_{10}$.

![Fig. 3. Two symbolic bisimilar automata](image)

The following theorem states that symbolic bisimulation completely captures the notion of $\nabla$-bisimulation.

**Theorem 1.** For $\upharpoonright$-closed $\phi$, $t \sim^\phi u$ iff $t \rho \sim^N u \rho$ for any $\rho \models \phi$

**Proof.** ($\Rightarrow$) Let $S$ be a symbolic bisimulation. Define

$$R = \{(t, D) \rho, (u, D) \rho) \mid \exists \phi : \rho \models \phi : (t, u, \phi) \in S \text{ and } D \subseteq A\}$$

12
We show that \( R \) is bisimulation. The fact that it is symmetric follows by symmetry of \( S \). In the following we suppose that \( ((t, D)\rho, (u, D)\rho) \in R \) as a consequence of \((t, u, \phi) \in S \) as indicated in (1), and prove the transfer property by doing case analysis on the type of transition.

discrete transition:

\[
(t, D)\rho \xrightarrow{a} (t', D')\rho' \\
\Rightarrow \{ \text{by } A1\gamma \} \\
\exists \gamma, \delta, x : t \xrightarrow{a, \gamma, \delta, x} t', D' = \emptyset, \rho' = \rho[x := 0], \text{ and } \rho \models \gamma \\
\Rightarrow \{ \text{by prop. 3 in Def. 6, since } (t, u, \phi) \in S \} \\
\exists \Phi : \Phi \text{ is a } (\phi \land \gamma)-\text{partition} : \forall \phi' \in \Phi : \\
\quad u \xrightarrow{a, \gamma', \delta', x'} u', \phi' \Rightarrow \gamma', (t', u', \phi' \downarrow_{xx \uparrow \epsilon}) \in S, \\
\]

By (1) \( \rho \models \phi \) and by (2) \( \rho \models \gamma \), hence \( \rho \models \phi \land \gamma \). Since \( \Phi \) is a \((\phi \land \gamma)\)-partition, then \( \rho \models \phi' \) for some \( \phi' \in \Phi \). Finally, since \( \phi' \Rightarrow \gamma' \) the \( \rho \models \gamma' \) also holds.

\[
\Rightarrow \{ \text{by observation} \} \\
\quad u \xrightarrow{a, \gamma', \delta', x'} u', \rho \models \gamma', \rho \models \phi', \text{ and } (t', u', \phi' \downarrow_{xx \uparrow \epsilon}) \in S \\
\Rightarrow \{ \text{by A1}\gamma, \text{ def. of } \downarrow_{xx'}, \text{ and Fact 1.4} \} \\
\quad (u, D)\rho \xrightarrow{a} (u', \emptyset)\rho[x' := 0], \\
\quad \rho[x' := 0] \models \phi' \downarrow_{xx \uparrow \epsilon}, \text{ and } (t', u', \phi' \downarrow_{xx \uparrow \epsilon}) \in S \\
\Rightarrow \{ \text{by (1)} \} \\
\quad (u, D)\rho \xrightarrow{a} (u', \emptyset)\rho[x' := 0] \text{ and } \\
\quad ((t', \emptyset)\rho[x' := 0], (u', \emptyset)\rho[x' := 0]) \in R \\
\Rightarrow \{ \text{by def. of reset} \} \\
\quad (u, D)\rho \xrightarrow{a} (u', \emptyset)\rho[x' := 0] \text{ and, } \\
\quad ((t', \emptyset)\rho[x := 0], (u', \emptyset)\rho[x' := 0]) \in R \\
\]

delay transition:

\[
(t, D)\rho \xrightarrow{d} (t', D')\rho' \\
\Rightarrow \{ \text{by } A2\gamma \} \\
\forall d' < d : \rho + d' \models -d(t, A - D), \text{ t } = t', \text{ D' } = D, \text{ and } \rho' = \rho + d \\
\]

By (1), \( \rho \models \phi \) for some \( \phi \) s.t. \( (t, u, \phi) \in S \). Moreover, by Fact 1.6, \( \rho + d' \models \phi \) for all \( d' \geq 0 \), in particular if \( d' < d \). As a consequence of prop. 4 in Def. 6, \( \rho + d' \models dl(t,A-D) \iff dl(u,A-D) \).

\[
\Rightarrow \{ \text{by observation} \} \\
\]

13
\[ \forall d' < d : \rho + d' \models -d(u, A - D) \]
\[ \Rightarrow \{ \text{by A2} \} \]
\[ (u, D)\rho \xrightarrow{d} (u, D)(\rho + d) \]
\[ \Rightarrow \{ \text{by (1), since } (t, u, \phi) \in S \text{ and } \rho + d \models \phi \text{ (see previous observation)} \} \]
\[ (u, D)\rho \xrightarrow{d} (u, D)(\rho + d) \text{ and } ((t, D)(\rho + d), (u, D)(\rho + d)) \in R \]

**drop transition:** Notice that both \((t, D)\rho \xrightarrow{\Sigma_E} (t, D \cup E)\rho\) and \((u, D)\rho \xrightarrow{\Sigma_E} (u, D \cup E)\rho\), by A3. Moreover, since \((t, u, \phi) \in S\) and \(\rho \models \phi\), by (1), \(((t, D \cup E)\rho, (u, D \cup E)\rho) \in R\).

**undrop transition:** Similarly, \((t, D)\rho \xrightarrow{\Delta} (t, \emptyset)\rho\) and \((u, D)\rho \xrightarrow{\Delta} (u, \emptyset)\rho\), by A4. Moreover, since \((t, u, \phi) \in S\) and \(\rho \models \phi\), by (1), \(((t, \emptyset)\rho, (u, \emptyset)\rho) \in R\).

\(\iff\) We prove that relation
\[ S = \{ (t, u, \phi) \mid \phi \text{ is } \uparrow \text{-closed and } \forall \psi \in \mathcal{RC}(\phi) : \exists \rho : \rho \models \psi : (t, \emptyset)\rho \sim (u, \emptyset)\rho \} \]

is a symbolic bisimulation. Since \(\sim\) is symmetric, \(S\) satisfies prop. 1 of Def. 6 and by definition, it satisfies prop. 2 as well. In the following we prove that \(S\) also satisfies properties 3 and 4 in Def. 6.

**Property 3:**

\[ t \xrightarrow{a, \gamma, \delta, x} t' \text{ and } (t, u, \phi) \in S \]
\[ \Rightarrow \{ \text{by (3)} \} \]
\[ t \xrightarrow{a, \gamma, \delta, x} t' \text{ and } \forall \psi \in \mathcal{RC}(\phi) : \exists \rho : \rho \models \psi : (t, \emptyset)\rho \sim (u, \emptyset)\rho \]

\[
\begin{align*}
&\text{Take } \Phi = \mathcal{RC}(\phi \land \gamma). \text{ Notice that it is a } (\phi \land \gamma)-\text{partition and that } \Phi \subseteq \\
&\mathcal{RC}(\phi) \text{ by Fact 1.2. Then } \psi \Rightarrow \gamma \text{ for all } \psi \in \Phi. \\
&\Rightarrow \{ \text{by observation} \} \\
&t \xrightarrow{a, \gamma, \delta, x} t' \text{ and } \forall \psi \in \Phi : \exists \rho : \rho \models \psi : (t, \emptyset)\rho \sim (u, \emptyset)\rho \land \rho \models \gamma \\
&\Rightarrow \{ \text{by A1} \} \\
&\forall \psi \in \Phi : \exists \rho : \rho \models \psi : \\
&(t, \emptyset)\rho \sim (u, \emptyset)\rho \text{ and } (t, \emptyset)\rho \xrightarrow{a} (t', \emptyset)\rho\{x := 0\} \\
&\Rightarrow \{ \text{is a bisimulation} \} \\
&\forall \psi \in \Phi : \exists \rho : \rho \models \psi : \\
&(u, \emptyset)\rho \xrightarrow{a} (u', D)\rho' \text{ and } (t', \emptyset)\rho\{x := 0\} \sim (u', D)\rho' \\
&\Rightarrow \{ \text{by A1} \} \\
&\forall \psi \in \Phi : \exists \rho : \rho \models \psi : \\
&u \xrightarrow{a, \gamma', \delta', x'} u', \rho \models \gamma', D' = \emptyset, \rho' = \rho\{x' := 0\}, \\
\text{and } (t', \emptyset)\rho\{x := 0\} \sim (u', \emptyset)\rho\{x' := 0\}
\end{align*}
\]
Since $\rho \models \psi$ and $\rho \models \gamma'$, $\psi \land \gamma'$ is not a contradiction and hence $\psi \Rightarrow \gamma'$.

⇒ {by observation}

∀$\psi \in \Phi : \exists \rho : \rho \models \psi : u \xrightarrow{a;\gamma';x'} u'$, $\psi \Rightarrow \gamma'$,
and $(t', \emptyset)\rho(x := 0) \sim (u', \emptyset)\rho(x' := 0)$
⇒ {by def. of reset}

∀$\psi \in \Phi : u \xrightarrow{a;\gamma';x'} u'$, $\psi \Rightarrow \gamma'$, and

∃$\rho : \rho[xx' := 0] \models \psi_{xx'} : (t', \emptyset)\rho[xx' := 0] \sim (u', \emptyset)\rho[xx' := 0]$
⇒ {by Lemma 3, since $\psi_{xx'}$ is a region by Fact 1.3}

∀$\psi \in \Phi : u \xrightarrow{a;\gamma';x'} u'$, $\psi \Rightarrow \gamma'$, and

∃$\xi \in RC(\psi_{xx'}) : \exists d \geq 0 : (\rho[xx' := 0] + d) \models \xi$
and $(t', \emptyset)\rho[xx' := 0] + d \sim (u', \emptyset)\rho[xx' := 0] + d$
⇒ {by Lemma 1 and logics}

∀$\psi \in \Phi : u \xrightarrow{a;\gamma';x'} u'$, $\psi \Rightarrow \gamma'$, and

∀$\xi \in RC(\psi_{xx'}) : \exists d \geq 0 : (\rho[xx' := 0] + d) \models \xi$
and $(t', \emptyset)\rho[xx' := 0] + d \sim (u', \emptyset)\rho[xx' := 0] + d$
⇒ {taking $\rho' = \rho[xx' := 0] + d$}

∀$\psi \in \Phi : u \xrightarrow{a;\gamma';x'} u'$, $\psi \Rightarrow \gamma'$, and

∀$\xi \in RC(\psi_{xx'}) : \exists \rho' : \rho' \models \xi$ and $(t', \emptyset)\rho' \sim (u', \emptyset)\rho'$
⇒ {by (3), since $\psi_{xx'}$ is $\models$-closed}

∀$\psi \in \Phi : u \xrightarrow{a;\gamma';x'} u'$, $\psi \Rightarrow \gamma'$, and $(t', u', \psi_{xx'}) \in S$

Property 4:

$(t, u, \phi) \in S$
⇒ {by (3)}

∀$\psi \in RC(\phi) : \exists \rho : \rho \models (t, \emptyset)\rho \sim (u, \emptyset)\rho$
⇒ {by Lemma 2}

∀$\psi \in RC(\phi) : \exists \rho : \rho \models \psi : \forall D \subseteq A : \rho \models dl(t, D) \leftrightarrow dl(u, D)$

$\rho \models \psi$ and $\rho \models dl(t, D) \leftrightarrow dl(u, D)$ implies $\psi \land (dl(t, D) \leftrightarrow dl(u, D))$ is not a contradiction.

⇒ {by Fact 1.2 and previous observation}
\[ \forall \psi \in \mathcal{RC}(\phi) : \forall D \subseteq A : \psi \Rightarrow (dl(t,D) \Leftrightarrow dl(u,D)) \]
\[ \Rightarrow \{ \text{by logics using the fact that } \phi \Leftrightarrow \bigvee \mathcal{RC}(\phi) \} \]
\[ \forall D \subseteq A : \phi \Rightarrow (dl(t,D) \Leftrightarrow dl(u,D)) \]

The following corollary specialises Theorem 1 to TADs and states that \(\sim^{\phi_0}\) symbolically characterises \(\sim\).

**Corollary 1.** Let \(\phi_0 \equiv \bigwedge_{x,y \in C_1 \cup C_2} (0 \leq x = y)\). \(T_1 \sim^{\phi_0} T_2\) if and only if \(T_1 \sim^\square T_2\).

5 The Coarsest Congruence Included in \(\sim\)

In this section, we show that \(\sim^{\phi_0}\) (and hence \(\sim^\square\), too) is the coarsest congruence for the parallel composition included in bisimulation. The first part of the section is devoted to prove that \(\sim^{\phi_0}\) is a congruence. It is interesting to notice that the proof of congruence is carried out fully at symbolic level (in contrast to the usual proof using the underlying transition system). To the best of our knowledge, this is a novel approach. In the second part we show that \(\sim^\square\) is the coarsest congruence included in \(\sim\).

The next two lemmas are required for the proof of congruence. Lemma 4 implies that a deadline of a set of actions can be decomposed as a disjunction of the deadlines of each of the actions. Lemma 5 states that if two locations \(t\) and \(u\) are symbolically bisimilar under a constraint \(\phi\), then a given action \(a\) is enabled in \(t\) if and only if it is enabled in \(u\) for all valuations that satisfy constraint \(\phi\).

**Lemma 4.** \(dl(s, D \cup E) \Leftrightarrow (dl(s, D) \vee dl(s, E))\)

**Proof.** Straightforward calculations.

**Lemma 5.** Define \(gd(s, a) = \bigvee \{ \gamma \mid s \xrightarrow{a;\gamma;\delta;\alpha} s' \text{ for some } \delta, \alpha, s' \}\). If \(S\) is a symbolic bisimulation such that \((t, u, \phi) \in S\), then \(\phi \Rightarrow (gd(t,a) \Leftrightarrow gd(u,a))\) is valid for all \(a \in A\).

**Proof.** Let \(S\) be a symbolic bisimulation with \((t, u, \phi) \in S\). By symmetry (property 1, Def. 6), it suffices to show that \(\phi \Rightarrow (gd(t,a) \Rightarrow gd(u,a))\). By definition of \(gd\), this follows by the claim that, for all \(\gamma\) such that \(t \xrightarrow{a;\gamma;\delta;\alpha} t'\), \(\phi \Rightarrow (\gamma \Rightarrow gd(u,a))\) (that is \((\phi \land \gamma) \Rightarrow gd(u,a))\) which is what we prove in the following.

\[ t \xrightarrow{a;\gamma;\delta;\alpha} t' \]
\[ \Rightarrow \{ \text{by prop. 3 in Def. 6, since } (t, u, \phi) \in S \} \]
\[ \exists \Phi : \Phi \text{ is a } (\phi \land \gamma)\text{-partition} : \forall \phi' \in \Phi : u \xrightarrow{a;\gamma';\delta';\alpha'} u' \text{ and } \phi' \Rightarrow \gamma' \]
\[ \Rightarrow \{ \gamma' \Rightarrow gd(u,a) \text{ by def. of } gd \} \]
\[ \exists \Phi : \Phi \text{ is a } (\phi \land \gamma)\text{-partition : } \forall \phi' \in \Phi : \phi' \Rightarrow gd(u, a) \]
\[ \Rightarrow \{((\phi \land \gamma) \Leftrightarrow \bigvee\Phi\} \]
\[ (\phi \land \gamma) \Rightarrow gd(u, a) \]
\[ \square \]

Now, we are in conditions to prove that \( \sim^\phi \) is a congruence for any parallel composition defined as in Section 2. In particular, we notice that the proof does not use constraints 1 and 4 imposed on \( \otimes \).

**Theorem 2.** Let \( T_1^j = (L_1^j, A_1^j, C_1^j, \rightarrow) \), for \( i, j \in \{1, 2\} \) such that \( C_1^i \cap C_1^j = \emptyset \) if \( i \neq k \) or \( j \neq l \). Then \( T_1^1 \sim^\phi T_1^2 \) and \( T_2^1 \sim^\phi T_2^2 \) imply \( T_1^1 \parallel B T_1^2 \sim^\phi B T_2^1 \parallel B T_2^2 \) for all \( B \in A \), operation \( \otimes \) and constraint \( \phi \).

**Proof.** Let \( S_1 \) and \( S_2 \) be two symbolic bisimulations that witness \( T_1^1 \sim^\phi T_1^2 \) and \( T_2^1 \sim^\phi T_2^2 \) respectively. Define
\[ S = \{(t_1, t_2), (u_1, u_2), \phi_1 \land \phi_2) \mid (t_1, u_1, \phi_1) \in S_1 \text{ and } (t_2, u_2, \phi_2) \in S_2\} \] (4)

We prove that \( S \) is also a symbolic bisimulation from which the theorem follows. For this, we check that \( ((t_1, t_2), (u_1, u_2), \phi_1 \land \phi_2) \in S \), obtained as in (4), satisfy the four properties in Def. 6. Property 1 is immediate since \( S_1 \) and \( S_2 \) also satisfy it. So is property 2: since \( \phi_1 \) and \( \phi_2 \) are \( \uparrow \text{-closed} \), so is \( \phi_1 \land \phi_2 \) using Fact 1.7. We proceed to check the remaining two properties.

**Property 3:** Suppose \( (t_1, t_2) \xrightarrow{a, \gamma, \delta, x} (t_1', t_2') \). Then three cases arise

**Case** \( (a \notin B \text{ with } t_1 \xrightarrow{a, \gamma, \delta, x} t_1' \text{ and } t_2' = t_2) \).
\[ t_1 \xrightarrow{a, \gamma, \delta, x} t_1' \Rightarrow \{ \text{by prop. 3 in Def. 6, since } (t_1, u_1, \phi_1) \in S_1 \} \]
\[ \exists \Phi : \Phi \text{ is a } (\phi_1 \land \gamma)\text{-partition :} \]
\[ \forall \phi \in \Phi : u_1 \xrightarrow{a, \gamma', \delta', x'} u_1', \phi \Rightarrow \gamma', \text{ and } (t_1', u_1', \phi \downarrow x_x \uparrow) \in S_1 \]
\[ \Rightarrow \{ \text{by Def. of } \parallel B \} \]
\[ \exists \Phi : \Phi \text{ is a } (\phi_1 \land \gamma)\text{-partition :} \]
\[ \forall \phi \in \Phi : (u_1, u_2) \xrightarrow{a, \gamma', \delta', x'} (u_1', u_2), \phi \Rightarrow \gamma', \]
\[ \text{ and } (t_1', u_1', \phi \downarrow x_x \uparrow) \in S_1 \]
\[ \Rightarrow \{ \text{by (4) and since } \phi \Rightarrow \gamma' \text{ implies } (\phi \land \phi_2) \Rightarrow \gamma' \} \]
\[ \exists \Phi : \Phi \text{ is a } (\phi_1 \land \gamma)\text{-partition :} \]
\[ \forall \phi \in \Phi : (u_1, u_2) \xrightarrow{a, \gamma', \delta', x'} (u_1', u_2), \phi \land \phi_2 \Rightarrow \gamma', \]
\[ \text{ and } (t_1', t_2), (u_1, u_2), (\phi \downarrow x_x \uparrow \land \phi_2) \in S \]

Notice that clocks in \( xx' \) are not manipulated by automata \( T_1^1 \) and \( T_2^1 \) and hence irrelevant in \( \phi_2 \). W.l.o.g. we therefore can assume that \( \phi_2 \Rightarrow \bigwedge \{x \geq 0 \mid x \in xx'\} \). Consequently \( \phi_2 \downarrow x_x \uparrow = \phi_2 \) since \( \phi_2 \) is \( \uparrow \text{-closed} \). (\( \dagger \))
\[ \Rightarrow \{ \text{Fact 1.7 and observation} \}

17
\[ \exists \Phi : \Phi \text{ is a} \ (\phi_1 \land \gamma)\text{-partition} : \]
\[ \forall \phi \in \Phi : \ (u_1, u_2) \xrightarrow{\alpha' \cdot \delta' \cdot x'} (u'_1, u_2), \ (\phi \land \phi_2) \Rightarrow \gamma', \]
\[ \text{and } ((t'_1, t_2), (u'_1, u_2), (\phi \land \phi_2)_{x'x'} \in S) \]

Take \( \Phi' = \{ \phi \land \phi_2 \mid \phi \in \Phi \} \). Then \( \Phi' \) is a \((\phi_1 \land \phi_2 \land \gamma)\)-partition.

\[ \Rightarrow \{ \text{by observation, taking } \phi' = \phi \land \phi_2 \} \]
\[ \exists \Phi' : \Phi' \text{ is a} \ (\phi_1 \land \phi_2 \land \gamma)\text{-partition} : \]
\[ \forall \phi' \in \Phi' : \ (u_1, u_2) \xrightarrow{\alpha' \cdot \delta' \cdot x'} (u'_1, u_2), \ \phi' \Rightarrow \gamma', \]
\[ \text{and } ((t'_1, t_2), (u'_1, u_2), (\phi'_{x'x'} \in S) \]

\textbf{Case} \((a \notin B \text{ with } t_2 \xrightarrow{\alpha' \cdot \delta \cdot x} t'_2 \text{ and } t'_1 = t_1)\). Symmetric to the previous case.

\textbf{Case} \((a \in B \text{ with } t_1 \xrightarrow{\alpha_1 \cdot \delta_1 \cdot x_1} t'_1, t_2 \xrightarrow{\alpha_2 \cdot \delta_2 \cdot x_2} t'_2, \gamma \equiv \gamma_1 \land \gamma_2, \text{ and } \delta \equiv (\delta_1, \gamma_1) \otimes (\delta_2, \gamma_2)\).

\[
\begin{align*}
\ & t_1 \xrightarrow{\alpha_1 \cdot \delta_1 \cdot x_1} t'_1 \text{ and } t_2 \xrightarrow{\alpha_2 \cdot \delta_2 \cdot x_2} t'_2 \\
\Rightarrow \ & \{ \text{by prop. 3 in Def. 6, since } (t_1, u_1, \phi_1) \in S_1 \text{ and } (t_2, u_2, \phi_2) \in S_2 \} \\
\ & \exists \Phi_1 : \Phi_1 \text{ is a } (\phi_1 \land \gamma_1)\text{-partition} : \\
\ & \forall \phi'_1 \in \Phi_1 : \ u_1 \xrightarrow{\alpha_1' \cdot \delta'_1 \cdot x'_1} u'_1, \ \phi'_1 \Rightarrow \gamma'_1, \text{ and } (t'_1, u'_1, \phi'_1_{x_1x'_1} \in S_1 \\
\ & \exists \Phi_2 : \Phi_2 \text{ is a } (\phi_2 \land \gamma_2)\text{-partition} : \\
\ & \forall \phi'_2 \in \Phi_2 : \ u_2 \xrightarrow{\alpha_2' \cdot \delta'_2 \cdot x'_2} u'_2, \ \phi'_2 \Rightarrow \gamma'_2, \text{ and } (t'_2, u'_2, \phi'_2_{x_2x'_2} \in S_2 \\
\Rightarrow \ & \{ \text{logics and notation} \} \\
\ & \exists \Phi_1, \Phi_2 : \Phi_i \text{ is a } (\phi_i \land \gamma_i)\text{-partition, for } i = 1, 2 : \\
\ & \forall \phi'_i \in \Phi_i, \ \phi'_j \in \Phi_j : \\
\ & u_1 \xrightarrow{\alpha_1' \cdot \delta'_1 \cdot x'_1} u'_1, \ \phi'_i \Rightarrow \gamma'_1, \ (t'_1, u'_1, \phi'_1_{x_1x'_1} \in S_1, \\
\ & u_2 \xrightarrow{\alpha_2' \cdot \delta'_2 \cdot x'_2} u'_2, \ \phi'_j \Rightarrow \gamma'_2, \text{ and } (t'_2, u'_2, \phi'_2_{x_2x'_2} \in S_2 \\
\ & \phi'_i \Rightarrow \gamma'_i \text{ and } \phi'_j \Rightarrow \gamma'_j \text{ imply } (\phi'_1 \land \phi'_2) \Rightarrow (\gamma'_1 \land \gamma'_2). \\
\ & \text{Besides, } (\phi'_1_{x_1x'_1} \land \phi'_2_{x_2x'_2} \in S_1, \phi'_1_{x_1x'_1} \land \phi'_2_{x_2x'_2} \in S_2 \text{ because of Fact 1.7} \\
\ & \text{and by observation (i) in previous case since clocks in } x_i x'_i \text{ do not appear in } \phi'_j \text{ for } i \neq j. \\
\Rightarrow \ & \{ \text{by def. of } ||^\circ_B, (4), \text{ and observation, taking } \delta' \equiv (\delta'_1, \gamma'_1) \otimes (\delta'_2, \gamma'_2) \} \\
\ & \exists \Phi_1, \Phi_2 : \Phi_i \text{ is a } (\phi_i \land \gamma_i)\text{-partition, for } i = 1, 2 : \\
\ & \forall \phi'_i \in \Phi_i, \ \phi'_j \in \Phi_j : \\
\ & (u_1, u_2) \xrightarrow{\alpha_1' \land \alpha_2' \cdot \delta' \cdot x_1x'_2} (u'_1, u_2), \ (\phi'_1 \land \phi'_2) \Rightarrow (\gamma'_1 \land \gamma'_2),
\end{align*}
\]
and \(((t'_1, t'_2), (u'_1, u'_2), (\phi'_1 \land \phi'_2) |_{x_1 x_2 x'_1 x'_2}) \in S\)

Take \(\Phi = \{ \phi'_1 \land \phi'_2 \mid \phi'_1 \in \Phi_1 \text{ and } \phi'_2 \in \Phi_2 \}\). Notice that \(\Phi\) is a \((\phi_1 \land \phi_2 \land \gamma_1 \land \gamma_2)\)-partition.

\(\Rightarrow\) \(\{\text{by observation, taking } \phi' = \phi'_1 \land \phi'_2\}\)

\(\exists \Phi: \Phi\) is a \((\phi_1 \land \phi_2 \land \gamma_1 \land \gamma_2)\)-partition :

\(\forall \phi' \in \Phi: (u_1, u_2) - a \cdot \gamma_1 \land \gamma_2 \cdot x_1 x_2 x'_1 x'_2 \to (u'_1, u'_2), \phi' \Rightarrow (\gamma'_1 \land \gamma'_2),\)

and \(((t'_1, t'_2), (u'_1, u'_2), \phi'_1 |_{x_1 x_2 x'_1 x'_2}) \in S\)

Property 4: We have to show that \(\phi_1 \land \phi_2 \Rightarrow (dl((t_1, t_2), A) \iff dl((u_1, u_2), A))\) for all \(A \subseteq A\). By Lemma 4, it suffices to prove that \((\phi_1 \land \phi_2) \Rightarrow (dl((t_1, t_2), \{a\}) \iff dl((u_1, u_2), \{a\}))\). Therefore, the following calculations are under the hypothesis that \(\phi_1 \land \phi_2\) holds. We consider two different cases.

Case \((a \notin B)\).

\(dl((t_1, t_2), \{a\})\)

\(\iff \{\text{def. of } dl\}\)

\(\vee \{\delta \mid (t_1, t_2) \xrightarrow{a \cdot \gamma \cdot x} (t'_1, t'_2) \text{ for some } \gamma, x, t'_1, t'_2\}\)

\(\iff \{\text{def. } \|m\text{ with } a \notin B\}\)

\(\vee \{\delta \mid t_1 \xrightarrow{a \cdot \gamma \cdot x} t'_1 \text{ for some } \gamma, x, t'_1\}\)

\(\vee \{\delta \mid t_2 \xrightarrow{a \cdot \gamma \cdot x} t'_2 \text{ for some } \gamma, x, t'_2\}\)

\(\iff \{\text{def. of } dl\}\)

\(dl(t_1, \{a\}) \vee dl(t_2, \{a\})\)

By (4), \((t_i, u_i, \phi_i) \in S_i\) from which \(\phi_i \Rightarrow (dl(t_i, \{a\}) \iff dl(u_i, \{a\}))\) by prop. 4 in Def. 6, for \(i = 1, 2\).

\(\iff \{\text{by observation, recalling that we assume } \phi_1 \land \phi_2\) holds\}

\(dl(u_1, \{a\}) \vee dl(u_2, \{a\})\)

\(\iff \{\text{reasoning as before}\}\)

\(dl((u_1, u_2), \{a\})\)

Case \((a \in B)\).

\(dl((t_1, t_2), \{a\})\)

\(\iff \{\text{def. of } dl\}\)

\(\vee \{\delta \mid (t_1, t_2) \xrightarrow{a \cdot \gamma \cdot x} (t'_1, t'_2) \text{ for some } \gamma, x, t'_1, t'_2\}\)

\(\iff \{\text{def. } \|m\text{ and } a \in B\}\)
\[ \nabla((\delta_1, \gamma_1) \otimes (\delta_2, \gamma_2)) \mid t_1 \xrightarrow{a;\gamma_1,\delta_1,x_1} t'_1 \text{ and } t_2 \xrightarrow{a;\gamma_2,\delta_2,x_2} t'_2 \text{ for some } x_1, x_2, t'_1, t'_2 \]

\[ \Leftrightarrow \{ \text{change of notation and logic} \}
\]

\[ \bigvee_{t_1} a;\gamma_1,\delta_1, x_1 \bigvee_{t'_1} a;\gamma_1,\delta_1, x_1 \bigvee_{t_2} a;\gamma_2,\delta_2, x_2 \bigvee_{t'_2} a;\gamma_2,\delta_2, x_2 \]

\[ \Leftrightarrow \{ \otimes \text{ distributes w.r.t. } \lor \}
\]

\[ \left( \bigvee_{t_1} a;\gamma_1,\delta_1, x_1 \right) \otimes \left( \bigvee_{t_2} a;\gamma_2,\delta_2, x_2 \right) \]

\[ \Leftrightarrow \{ \text{def. of } dl \text{ and } gd \}
\]

\[ (dl(t_1, \{a\}), gd(t_1, a)) \otimes (dl(t_2, \{a\}), gd(t_2, a)) \]

By (4), \((t_i, u_i, \phi_i) \in S_i \) from which \( \phi_i \Rightarrow (dl(t_i, \{a\}) \Leftrightarrow dl(u_i, \{a\})) \) by prop. 4 in Def. 6, and \( \phi_i \Rightarrow (gd(t_i, a) \Leftrightarrow gd(u_i, a)) \) by Lemma 5, for \( i = 1, 2. \)

\[ \Leftrightarrow \{ \text{by observation and cond. 2 of } \otimes \text{, recalling that we assume } \phi_1 \land \phi_2 \}
\]

\[ (dl(u_1, \{a\}), gd(u_1, a)) \otimes (dl(u_2, \{a\}), gd(u_2, a)) \]

\[ \Leftrightarrow \{ \text{reasoning as before} \}
\]

\[ dl((u_1, u_2), \{a\}) \]

\[ \square \]

Because of Corollary 1 we have the next corollary of Theorem 2.

**Corollary 2.** \( \sim^\nabla \) is a congruence for parallel composition.

The next lemma is core for the proof that \( \sim^\nabla \) is the coarsest congruence included in \( \sim. \) We notice that it does not use constraints 1, 2, and 3 imposed on \( \otimes. \) The lemma exhibits a test automata \( T_t \) that distinguish, modulo bisimulation, two automata that are not \( \nabla \)-bisimilar. Automata \( T_t \) is built by adding extra actions in such a way that, when composed with an automata \( T \), the composition can mimic in the original semantics the behaviour of \( T \) in the extended semantics. In fact, the extra actions are the same drop (\( \nabla_D \)) and undrop (\( \Delta \)) actions of the extended semantics.

**Lemma 6.** Define the test automata \( T_t \) with set of locations \( L_t = \{s_D \mid D \subseteq \mathcal{A}\}; \emptyset_t = \{s_\emptyset\}; \) set of clocks \( C_t = \emptyset; \) set of actions \( \mathcal{A} \cup \nabla \cup \{\Delta\} \) and for all \( D, D' \subseteq \mathcal{A}, a \notin D \) define

\[ s_D \xrightarrow{a;tt,0_\emptyset} s_\emptyset \quad s_D \xrightarrow{\nabla_{D'}, tt,f} s_{D \cup D'} \quad s_D \xrightarrow{\Delta, tt,f} s_\emptyset \]

20
Let $T_1$ and $T_2$ be TADs with set of locations $\mathcal{L}_1$ and $\mathcal{L}_2$ respectively. Suppose that $T_1 \parallel_A T_1 \sim T_2 \parallel_A T_2$. Then, relation

$$R = \{(t_1, D) \rho_1, (t_2, D) \rho_2 \mid t_1 \in \mathcal{L}_1, \ t_2 \in \mathcal{L}_2, \ s_D \in \mathcal{L}_t, \ \text{and} \ (t_1, s_D) \rho_1 \sim (t_2, s_D) \rho_2 \}$$

is a bisimulation that witness $T_1 \sim ^\bigtriangleup T_2$.

**Proof.** First notice that for all initial location $t_1^0$ of $T_1$ there is an initial location $t_2^0$ of $T_2$ such that $(t_1^0, \emptyset)(C_1 \mapsto 0) \sim (t_2^0, \emptyset)(C_2 \mapsto 0)$. Then $(T_1, \emptyset)(C_1 \mapsto 0), (T_2, \emptyset)(C_2 \mapsto 0)) \in R$. Similarly, we have that for all initial location $t_1^0$ of $T_2$ there is an initial location $t_1^0$ of $T_1$ such that $((t_1^0, \emptyset)(C_1 \mapsto 0), (t_2^0, \emptyset)(C_2 \mapsto 0)) \in R$. Then, provided $R$ is a bisimulation, $T_1 \sim ^\bigtriangleup T_2$.

Notice, besides, that $R$ is symmetric by symmetry of $\sim$. We proceed to prove the transfer property by doing case analysis on the type of edge.

discrete transition:

$$(t_1, D) \rho_1 \overset{a}{\rightarrow} (t'_1, D') \rho'_1$$

$$\Rightarrow \{ \text{by A1}\}$$

$$\exists \gamma_1, \delta_1, x_1 : t_1 \overset{a, \gamma_1, \delta_1, x_1}{\rightarrow} t'_1, \ D' = \emptyset, \ \rho'_1 = \rho_1\{x_1 := 0\}, \ \text{and} \ \rho_1 \models \gamma_1$$

$$\Rightarrow \{ \text{by def. of } \parallel_A \text{ and def. of } T_i \}$$

$$\exists \gamma_1, \delta_1, x_1 : (t_1, s_D) \xrightarrow{\Delta t, \mathbf{f}, \emptyset} (t_1, s_\emptyset) \xrightarrow{a, \gamma_1, \delta_1, \gamma_1 \otimes (0, t_1), x_1} (t'_1, s_\emptyset)$$

$$\text{and} \ \rho_1 \models \gamma_1$$

$$\Rightarrow \{ \text{by A1} \}$$

$$(t_1, s_D) \rho_1 \xrightarrow{\Delta} (t_1, s_\emptyset) \rho_1 \xrightarrow{a} (t'_1, s_\emptyset) \rho_1\{x_1 := 0\}$$

$$(t_2, s_D) \rho_2 \xrightarrow{\Delta} (t_2, s_D') \rho_2 \xrightarrow{a} (t'_2, s_{D'}) \rho_2$$

$$\Rightarrow \{ \text{by A1}_\mathbf{f} \}$$

$$\exists \gamma, \delta, x : (t_2, s_D) \xrightarrow{\Delta \gamma, \delta, x} (t''_2, s_{D''}), \ \rho_2 \models \gamma, \ \rho''_2 = \rho_2\{x := 0\},$$

$$(t''_2, s_{D''}) \rho_2\{x := 0\} \xrightarrow{a} (t'_2, s_{D''}) \rho_2,$$

$$\text{and} \ (t'_1, s_\emptyset) \rho_1\{x_1 := 0\} \xrightarrow{a} (t'_2, s_{D''}) \rho_2$$

$$\Rightarrow \{ \gamma = \mathbf{tt}, \ \delta = \mathbf{ff}, \ x = \emptyset, \ D'' = \emptyset, \ \text{and} \ t''_2 = t_2, \ \text{by def. of } \parallel_A \text{ and } T_i \}$$

$$(t_2, s_D) \xrightarrow{\Delta t, \mathbf{f}, \emptyset} (t_2, s_\emptyset),$$

$$(t_2, s_\emptyset) \rho_2 \xrightarrow{a} (t'_2, s_{D''}) \rho_2,$$

and $$(t'_1, s_\emptyset) \rho_1\{x_1 := 0\} \sim (t'_2, s_{D''}) \rho_2$$.  

21
\( \Rightarrow \{ \text{by A1}_\gamma \} \)

\[ \exists \gamma_2, \delta_2, x_2 : (t_2, s_{\emptyset}) \xrightarrow{a, \gamma_2, \delta_2, x_2} (t_2', s_D'), \rho_2' = \rho_2\{x_2 := 0\}, \]

\( \rho_2 \models \gamma_2, \) and \((t_1', s_{\emptyset})_{\rho_1}\{x_1 := 0\} \sim (t_2', s_D')_{\rho_2}\{x_2 := 0\} \)

\[ \Rightarrow \{ \text{by def. of } A_\gamma \text{ and def. of } T_t \} \]

\[ \exists \gamma_2, \delta_2, x_2 : t_2 \xrightarrow{a, \gamma_2, \delta_2, x_2} t_2', D'' = \emptyset, \rho_2 \models \gamma_2, \] and \((t_1', s_{\emptyset})_{\rho_1}\{x_1 := 0\} \sim (t_2', s_{\emptyset})_{\rho_2}\{x_2 := 0\} \)

\[ \Rightarrow \{ \text{by A1}_\gamma \text{ and (5)} \} \]

\[ (t_2, D)_{\rho_2} \xrightarrow{a} (t_2', \emptyset)_{\rho_2}\{x_2 := 0\}, \] and \((t_1', \emptyset)_{\rho_1}\{x_1 := 0\} \sim (t_2', \emptyset)_{\rho_2}\{x_2 := 0\} \)

**delay transition:** We first notice that

\[ -d(t, A - D) = -\bigvee \{ \delta : t \xrightarrow{a, \gamma, \delta, x} t' \text{ and } a \in A - D \text{ for some } \gamma, x, t' \} \]

Recall that \( s_D \xrightarrow{a, \gamma', \delta', x'} s' \) implies \( a \in A - D, \gamma' = tt, \delta' = 0_\delta, \) and \( s' = s_{\emptyset}, \) and that \((0_\delta, tt)\) is neutral for \( \emptyset. \) By def. of \( A_\gamma \) we obtain:

\[ = -\bigvee \{ (t, s_D) \xrightarrow{a, \gamma, \delta, x} (t', s_{\emptyset}) \text{ for some } \gamma, x, t' \} \]

\[ = tp_a(t, s_D) \tag{6} \]

Now we calculate:

\[ (t_1, D)_{\rho_1} \xrightarrow{d} (t_1', D')_{\rho_1'} \]

\[ \Rightarrow \{ \text{by A2}_\gamma \} \]

\[ \forall d' < d : \rho_1 + d' \models -d(t_1, A - D), \ t_1' = t_1, \ D' = D, \text{ and } \rho_1' = \rho_1 + d \]

\[ \Rightarrow \{ \text{by (6)} \} \]

\[ \forall d' < d : \rho_1 + d' \models tp_a(t_1, s_D) \text{ and } \rho_1' = \rho_1 + d \]

\[ \Rightarrow \{ \text{by A2} \} \]

\[ (t_1, s_D)_{\rho_1} \xrightarrow{d} (t_1, s_D)_{(\rho_1 + d)} \]

\[ \Rightarrow \{ (t_1, s_D)_{\rho_1} \sim (t_2, s_D)_{\rho_2} \} \]

\[ (t_2, s_D')_{\rho_2} \xrightarrow{d} (t_2', s_D')_{\rho_2'} \text{ and } (t_1, s_D)_{(\rho_1 + d)} \sim (t_2, s_D)_{(\rho_2 + d)} \]

\[ \Rightarrow \{ \text{by A2} \} \]

\[ \forall d' < d : \rho_2 + d' \models tp_a(t_2, s_D), \ t_2' = t_2, \ D'' = D, \rho_2' = \rho_2 + d, \]

and \((t_1, s_D)_{(\rho_1 + d)} \sim (t_2, s_D)_{(\rho_2 + d)} \)

\[ \Rightarrow \{ \text{by (6)} \} \]

\[ \forall d' < d : \rho_2 + d' \models -d(t_2, D) \text{ and } (t_1, s_D)_{(\rho_1 + d)} \sim (t_2, s_D)_{(\rho_2 + d)} \]

\[ \Rightarrow \{ \text{by A2}_\gamma \text{ and (5)} \} \]

\[ (t_2, D)_{\rho_2} \xrightarrow{d} (t_2, D)_{(\rho_2 + d)} \text{ and } (t_1, D)_{(\rho_1 + d)} \sim (t_2, D)_{(\rho_2 + d)} \]
drop transition:

\[(t_1, D)\rho_1 \xrightarrow{\nabla_E} (t_1', D')\rho_1'
\]

\[\Rightarrow \{\text{by A3}\}\]
\[D' = D \cup E, \ \rho_1' = \rho_1, \ \text{and} \ t_1' = t_1\]

\[\Rightarrow \{\text{by def. of } ||_\land^\bot \text{ and def. of } T_t\}\]
\[(t_1, s_D) \xrightarrow{\nabla_E, \mathrm{tt}, \mathrm{ff}, \mathrm{e}} (t_1, s_{D \cup E})\]

\[\Rightarrow \{\text{by A2}\}\]
\[(t_1, s_D)\rho_1 \xrightarrow{\nabla_E} (t_1, s_{D \cup E})\rho_1\]

\[\Rightarrow \{(t_1, s_D)\rho_1 \sim (t_2, s_D)\rho_2\}\]
\[(t_2, s_D)\rho_2 \xrightarrow{\nabla_E} (t_2', s_{D'})\rho_2' \text{ and } (t_1, s_{D \cup E})\rho_1 \sim (t_2', s_{D'})\rho_2'\]

\[\Rightarrow \{\text{by A2, def. of } ||_\land^\bot \text{ and def. of } T_t\}\]
\[(t_2, s_D) \xrightarrow{\nabla_E, \mathrm{tt}, \mathrm{ff}, \mathrm{e}} (t_2, s_{D \cup E})\]
\[t_2' = t_2, \ D' = D \cup E, \ \rho_2' = \rho_2, \text{ and } (t_1, s_{D \cup E})\rho_1 \sim (t_2, s_{D \cup E})\rho_2\]

\[\Rightarrow \{\text{by A3 and (5)}\}\]
\[(t_2, D)\rho_2 \xrightarrow{\nabla_E} (t_2, D \cup E)\rho_2 \text{ and } (t_1, D \cup E)\rho_1 \sim (t_2, D \cup E)\rho_2\]

undrop transition:

\[(t_1, D)\rho_1 \xrightarrow{\Delta} (t_1', D')\rho_1'
\]

\[\Rightarrow \{\text{by A4}\}\]
\[D' = \emptyset, \ \rho_1' = \rho_1, \ \text{and} \ t_1' = t_1\]

\[\Rightarrow \{\text{by def. of } ||_\land^\bot \text{ and def. of } T_t\}\]
\[(t_1, s_D) \xrightarrow{\Delta, \mathrm{tt}, \mathrm{ff}, \mathrm{e}} (t_1, s_\emptyset)\]

\[\Rightarrow \{\text{by A2}\}\]
\[(t_1, s_D)\rho_1 \xrightarrow{\Delta} (t_1, s_\emptyset)\rho_1\]

\[\Rightarrow \{(t_1, s_D)\rho_1 \sim (t_2, s_D)\rho_2\}\]
\[(t_2, s_D)\rho_2 \xrightarrow{\Delta} (t_2', s_{D'})\rho_2' \text{ and } (t_1, s_\emptyset)\rho_1 \sim (t_2', s_{D'})\rho_2'\]

\[\Rightarrow \{\text{by A2, def. of } ||_\land^\bot \text{ and def. of } T_t\}\]
\[(t_2, s_D) \xrightarrow{\Delta, \mathrm{tt}, \mathrm{ff}, \mathrm{e}} (t_2, s_\emptyset)\]
\[t_2' = t_2, \ D' = \emptyset, \ \rho_2' = \rho_2, \text{ and } (t_1, s_\emptyset)\rho_1 \sim (t_2, s_\emptyset)\rho_2\]

\[\Rightarrow \{\text{by A4 and (5)}\}\]
\[(t_2, D)\rho_2 \xrightarrow{\Delta} (t_2, \emptyset)\rho_2 \text{ and } (t_1, \emptyset)\rho_1 \sim (t_2, \emptyset)\rho_2\]

\[\square\]

The fact that \(\sim^\land\) and \(\sim^E\) are the coarsest congruence included in \(\sim\) follows from the previous lemma:
Theorem 3. Fix an operation $\circ$ satisfying conditions 1 and 2 in Section 2. $\sim^\nabla$ and hence $\sim^\phi_0$ are the coarsest congruence for the family of parallel composition $\|_B^\circ, B \subseteq A$ included in $\sim$.

Proof. Define $\sim^\circ$ to be the coarsest congruence for parallel composition contained in $\sim$, that is $T_1 \sim^\circ T_2 \iff \forall T, B : T_1 \|_B^\circ T \sim T_2 \|_B^\circ T$. We show that $\sim^\nabla = \sim^\phi_0 = \sim^\circ$.

The fact that $\sim^\nabla = \sim^\phi_0 \subseteq \sim^\circ \subseteq \sim$ follows from Lemma 1, Corollary 1, Theorem 2 and the fact that $\sim^\circ$ is the coarsest congruence included in $\sim$.

On the other direction, that is $\sim^\circ \subseteq \sim^\phi_0 \subseteq \sim$, notice that if $T_1 \sim^\circ T_2$ implies $T_1 \|_A^\circ T_1 \sim T_2 \|_A^\circ T_2$ with $T_1$ as in Lemma 6. Using Lemma 6 we can conclude that $T_1 \sim^\nabla T_2$.

6 Concluding Remarks

On Deciding $\nabla$-bisimulation. Our symbolic characterisation is based on the works of Lin & Yi [21] and Cerans [12]. In particular, [12] states that bisimulation is decidable for timed automata. The same applies to our relation. Since the number of regions is finite so is the number of (relevant) constraints (modulo logic equivalence) and as a consequence also the number of relevant $\uparrow$-closed constraints. Therefore, any possible symbolic bisimulation relating two TADs will also be finite. Besides, operations $\downarrow_\phi$ and $\uparrow_\phi$ are expressible in terms of constraints, and it is possible to decide validity of the constraints on clocks. Following [12], checking that two TADs $T_1$ and $T_2$ are $\nabla$-bisimilarity is then possible by taking relation $S = \{(t, u, \psi) \mid \phi \in R(C(tt)) \}$ (which is the finest partition possible since $R(C(tt)$ is the set of all regions) and checking that the transfer rules in Def. 6 hold for all tuples reachable from some set $I \subseteq (S \cap (ini_1 \times ini_2 \times S(\phi_0)))$ such that it relates all initial states of $T_1$ (resp. $T_2$) with some initial state of $T_2$, (resp. $T_1$).

Of course, this approach to decide $\nabla$-bisimulation is very expensive (the number of region is exponential in the number of clocks [2]). Smarter approaches could be achieved following ideas of e.g. [1].

A Remark on Symbolic Bisimulation. The third constraint in the definition of symbolic bisimulation (Def. 6) can be relaxed as follows:

whenever $t \xrightarrow{a,\gamma,\delta,\chi} t'$, there is a $(\phi \land \gamma)$-partition $\Phi$ such that for each $\phi' \in \Phi$, $u \xrightarrow{a',\delta',\chi'} u'$, $\phi' \Rightarrow \gamma'$, $\phi'|_{S(\psi)} \Rightarrow \psi$, and $(t', u', \psi) \in S$, for some $\psi$, $\gamma'$, $\delta'$, $\chi$ and $u'$.

the difference being on the existence of $\psi$ such that $\phi'|_{S(\psi)} \Rightarrow \psi$. It is not difficult to check that the new characterisation is equivalent to the original definition. This modification is important since it allows to obtain smaller relations due to the fact that a tuple $(t, u, \phi) \in S$ is redundant if there is a different tuple $(t, u, \phi') \in S$ such that $\phi \Rightarrow \phi'$.
On Synchronising Constraints in Parallel Compositions. In [6] the synchronisation of guards and deadlines of synchronising actions are defined by two operations which we call here $\oplus$ and $\otimes$ respectively. Some conditions are imposed in $\oplus$ and the only condition imposed in $\otimes$ is that $(d_1, \gamma_1) \otimes (d_2, \gamma_2) \Rightarrow (\gamma_1 \lor \gamma_2) \\land (\gamma_1 \lor \gamma_2)$ whenever $d_1 \Rightarrow \gamma_1$ and $d_2 \Rightarrow \gamma_2$ ([6] also suggest that $(d_1, \gamma_1) \otimes (d_2, \gamma_2) \Rightarrow (d_1 \lor d_2) \\land (d_1 \lor d_2)$ should hold). We will only discuss here some particular examples that have recurred on the works of Sifakis et al. (see, e.g. [7, 5, 6]). We first focus on the guard:

$\oplus = \land$. This is the one we use and amounts to check that both guards are enables in order to enable the synchronised transition.

$\oplus = \lor$. The synchronised transition is able to execute if any of the partners is able to do it.

$\oplus = \max$, where $\gamma_1 \max \gamma_2 = (\gamma_1 \land \gamma_2) \lor (\gamma_2 \land \gamma_1)$. In this case, a component is willing to synchronise if the synchronising transition was enabled in the past and the other component is ready to synchronise now.

$\oplus = \min$, where $\gamma_1 \min \gamma_2 = (\gamma_1 \land \gamma_2) \lor (\gamma_2 \land \gamma_1)$ with $\downarrow$ being the time predecessor operator (the dual of $\uparrow$). In this case, the synchronised guard anticipates the execution of the synchronising transitions.

Our congruence relation only works for $\land$. It is debatable how reasonable are the other operations. Synchronisation through $\lor$ is highly questionable. It is expected that automata $T_{11}$ and $T_{12}$ in Fig. 4 are equivalent under any reasonable criterion. Nevertheless, the composition $T_{11} \mid_\oplus T''$ can perform action $a$ at any moment while $T_{12} \mid_\oplus T''$ cannot.

Under $\min$, a component may anticipate the future behaviour of the synchronising partner. In [7] and [6], the authors suggest that the intention of this synchronisation is that the earliest synchronising transition makes irrelevant the second one (e.g. a tramway leaves a crossing and after a while it signals to allow the change of the traffic light though it may be ignored if the light has already changed [6]). This intuition does not completely match the behaviour of $\min$ which will speed up the slower component allowing it to do activity otherwise impossible. This is observed when automata $T'''$ is composed with $T_{13}$ and with $T_{14}$ synchronising on $a$ (see Fig. 4). Notice that $T_{13}$ and $T_{14}$ exhibit an apparent equal behaviour since action $T_{13}$ always arrive too late to execute action $b$. However, the composition $T_{13} \mid_\oplus T'''$ may hasten the synchronisation on $a$ so that $b$ can be executed.

![Fig. 4. $T_{11} \sim^\forall T_{12}$, $T_{13} \sim^\forall T_{14}$, and $T_{15} \sim^\forall T_{16}$](image-url)

25
Dually, under max, an automata may allow the execution of the synchronising action if it was enabled in the past. Notice that \( T_{15} \) and \( T_{16} \) in Fig. 4 exhibit equivalent behaviour: \( c \) cannot be executed in \( T_{15} \) since clock \( y \) is always set too early. Instead, the composition with \( T'' \) synchronising on \( a \) will delay the execution long enough as to set \( y \) sufficiently late to enable the \( c \) transition. The intention behind this form of synchronisation is that the fastest component can always wait for the slowest. This design choice seems an adequate choice to use with soft deadlines. Notice also that the appearance of new activity seems reasonable since it may be important to cope with the occasional delay. What is debatable is the need of max since this type of synchronisation can easily be represented using \( \land \): Notice that the max synchronisation does not allow any test automata to distinguish between \( ° \) and \( °^* \). Hence, it is probably more reasonable to model this kind of synchronisation using \( \land \) instead of max and let all guards be \( \uparrow \)-closed.

With respect to deadlines, [6] is more liberal. The two type of synchronising deadlines that stand out are:

- **Patient synchronisation**: \((\delta_1, \gamma_1) \otimes (\delta_2, \gamma_2) = \delta_1 \land \delta_2 \) with \( 0_\delta = \text{tt} \), and
- **Impatient synchronisation**: \((\delta_1, \gamma_1) \otimes (\delta_2, \gamma_2) = (\delta_1 \lor \delta_2) \land (\gamma_1 \land \gamma_2) \) with \( 0_\delta = \text{ff} \).

The nomenclature corresponds to [15] but these definitions are already introduced in [24] with the names of flexible and stiff respectively. Patient synchronisation allows to model soft deadlines, in the sense that one of the components is always willing to wait for the other (as long as its guards remain valid). On the other hand, impatient synchronisation impose urgency and obliges the execution as soon as both partners are ready to execute the synchronising transition. Both [24] and [15] give a weaker definition of impatient synchronisation: \((\delta_1, \gamma_1) \otimes (\delta_2, \gamma_2) = \delta_1 \lor \delta_2 \). Taking \( 0_\delta = \text{ff} \), our result is also valid for this definition. The only problem with it is that it does not preserve time reactivity, i.e. condition 1 on \( \otimes \) (see Sec. 2) does not hold.

We finally mention that \( \neg \)-bisimulation is still a congruence for \( \|\|_\rho \) if condition 4 on \( \otimes \) is dropped. However, it is *not* the coarsest congruence in \( \sim \) any longer. (This can easily be seen by taking \((\delta_1, \gamma_1) \otimes (\delta_2, \gamma_2) = \text{ff} \).)

**Conclusions.** We have characterised the coarsest congruence for parallel compositions of TADs with soft and hard deadline synchronisation that is included in bisimulation. We also gave a symbolic characterisation of it and show that it is decidable. An aside novelty in our result is that the proof of congruence was entirely carried out in the symbolic semantics rather than resorting to the underlying transition system. The choice on this strategy is not fortuitous. It is mainly due to the complexity on defining an equivalent parallel composition on transition systems. To begin with, any possible definition needs to be tailored for a particular choice of deadline. Besides, it would need complex bookkeeping to know which possible deadline is blocking the passage of time. Many other different complications appear depending on the choice of \( \otimes \).

We finally discussed different types of synchronisation in parallel composition and conclude that our choice is both reasonable and sufficiently expressive as to consider the modelling of both soft and hard real-time constraints.
Acknowledgments. We thank Frits Vaandrager for his remarks on early drafts that helped to improve the quality of the paper.

References