Avoiding Shared Clocks in Networks of Timed Automata

Sandie Balaguer, Thomas Chatain

Avoiding Shared Clocks in Networks of Timed Automata

Sandie Balaguer, Thomas Chatain

RESEARCH REPORT
N° 7990
June 2012
Project-Team MExICo

ISSN 0249-6399 ISRN INRIA/RR--7990--FR+ENG

Avoiding Shared Clocks in Networks of Timed Automata

Sandie Balaguer, Thomas Chatain*

Project-Team MExICo

Research Report n° 7990 — June 2012 — 23 pages

Abstract: Networks of timed automata (NTA) are widely used to model distributed real-time systems. Quite often in the literature, the automata are allowed to share clocks, i.e. the transitions of one automaton may be guarded by a condition on the value of clocks reset by another automaton. This is a problem when one considers implementing such model in a distributed architecture, since reading clocks a priori requires communications which are not explicitly described in the model. We focus on the following question: given a NTA $A_1 \parallel A_2$ where A_2 reads some clocks reset by A_1, does there exist a NTA $A_1' \parallel A_2'$ without shared clocks with the same behavior as the initial NTA?

For this, we allow the automata to exchange information during synchronizations only, in particular by copying the value of their neighbor's clocks.

We discuss a formalization of the problem and give a criterion using the notion of contextual timed transition system, which represents the behavior of A_2 when in parallel with A_1. Finally, we effectively build $A_1' \parallel A_2'$ when it exists.

Key-words: networks of timed automata, shared clocks, implementation on distributed architecture, contextual timed transition system, behavioral equivalence for distributed systems

* This work is partially supported by the French ANR project ImpRo.

RESEARCH CENTRE
SACLAY – ÎLE-DE-FRANCE

Parc Orsay Université
4 rue Jacques Monod
91893 Orsay Cedex

S'affranchir des horloges partagées dans les réseaux d'automates temporisés

Résumé : Les réseaux d'automates temporisés sont largement utilisés dans la modélisation des systèmes temps-réel distribués. Le plus souvent dans la littérature, le partage d'horloges entre les différents automates est autorisé : les transitions d'un automate peuvent être conditionnées par la valeur d'horloges remises à zéro par un autre automate. Cela pose problème lorsque l'on envisage l'implantation d'un tel modèle sur une architecture distribuée, puisque la lecture des horloges requiert a priori des communications qui ne sont pas décrites explicitement dans le modèle. Nous nous intéressons à la question suivante : étant donné un réseau d'automates temporisés $A_1 \parallel A_2$ où A_2 lit des horloges remises à zéro par A_1, existe-t-il un réseau d'automates temporisés $A'_1 \parallel A'_2$ sans horloges partagées avec le même comportement que le réseau initial ?

Dans cette optique, nous autorisons les automates à échanger de l'information pendant les synchronisations seulement, en copiant les valeurs des horloges de leur voisin.

Nous discutons d'abord d'une formalisation de ce problème, puis nous donnons un critère pour décider de l'existence du système sans horloges partagées, en introduisant la notion de système de transition temporisé contextuel. Ce système de transition représente le comportement de A_2 lorsqu'il est en parallèle avec A_1. Enfin, nous montrons comment construire $A'_1 \parallel A'_2$ quand il existe.

Mots-clés : réseaux d'automates temporisés, horloges partagées, implantation sur une architecture distribuée, système de transition temporisé contextuel, équivalence de comportement pour les systèmes distribués

1 Introduction

Timed automata [AD94] are one of the most famous formal models for real-time systems. They have been deeply studied and very mature tools are available, like Uppaal [LPY97], Epsilon [CGL93] and Kronos [BDM+98].

Networks of Timed Automata (NTA) are a natural generalization to model real-time distributed systems. In this formalism each automaton has a set of clocks that constrain its real-time behavior. But quite often in the literature, the automata are allowed to share clocks, which provides a special way of making the behavior of one automaton depend on what the others do. Actually shared clocks are relatively well accepted and can be a convenient feature for modeling systems. Moreover, since NTA are almost always given a sequential semantics, shared clocks can be handled very easily even by tools: once the NTA is transformed into a single timed automaton by the classical product construction, the notion of distribution is lost and the notion of shared clock itself becomes meaningless. Nevertheless, implementing a model with shared clocks in a distributed architecture is not straightforward since reading clocks a priori requires communications which are not explicitly described in the model.

Here we are concerned with the expressive power of shared clocks according to the distributed nature of the system. We are not aware of any previous study about this aspect. Our purpose is to identify NTA where sharing clocks could be avoided, i.e. NTA which syntactically use shared clocks, but whose semantics could be achieved by another NTA without shared clocks. To simplify, we look at NTA made of two automata A_1 and A_2 where only A_2 reads clocks reset by A_1. The first step is to formalize what aspect of the semantics we want to preserve in this setting. Then the idea is essentially to detect cases where A_2 can avoid reading a clock because its value does not depend on the actions that are local to A_1 and thus unobservable to A_2. To generalize this idea we have to compute the knowledge of A_2 about the state of A_1. We show that this knowledge is maximized if we allow A_1 to communicate its state to A_2 each time they synchronize on a common action.

In order to formalize our problem we need an appropriate notion of behavioral equivalence between two NTA. We explain why classical comparisons based on the sequential semantics, like timed bisimulation, are not sufficient here. We need a notion that takes the distributed nature of the system into account. That is, a component cannot observe the moves and the state of the other and must choose its local actions according to its partial knowledge of the state of the system. We formalize this idea by the notion of contextual timed transition systems (contextual TTS).

Then we express the problem of avoiding shared clocks in terms of contextual TTS and we give a characterization of the NTA for which shared clocks can be avoided. Finally we effectively construct a NTA without shared clocks with the same behavior as the initial one, when this is possible. A possible interest is to allow a designer to use shared clocks as a high-level feature in a model of a protocol, and rely on our transformation to make it implementable.

Related work. The semantics of time in distributed systems has already been debated. The idea of localizing clocks has already been proposed and some authors [ABG+08, DL07, BJLY98] have even suggested to use local-time semantics with independently evolving clocks. Here we stay in the classical setting of perfect clocks evolving at the same speed. This is a key assumption that provides an implicit synchronization and lets us know some clock values without reading them.

Many formalisms exist for real-time distributed systems, among which NTA [AD94] and time Petri nets [MF76]. So far, their expressiveness was compared [BCH+05, BR08, CR06, Srb08] essentially in terms of sequential semantics that forget concurrency. In [BCH12], we defined a concurrency-preserving translation from time Petri nets to networks of timed automata.

While partial-order semantics and unfoldings are well known for untimed systems, they have been very little studied for distributed real-time systems [CCJ06, BHR06]. Partial order reductions for (N)TA were proposed in [Min99, BJLY98, LNZ05]. Behavioral equivalence relations for distributed systems, like history-preserving bisimulations were defined for untimed systems only [BDKP91, vGG01].

Finally, our notion of contextual TTS deals with knowledge of agents in distributed systems. This is the aim of epistemic logics [HFMV95], which have been extended to real-time in [WL04, Dim09]. Our notion of contextual TTS also resembles the technique of partitioning states based on observation, used in timed games with partial observability [BDMP03, DLLN09].

Organization of the paper. The paper is organized as follows. Section 2 recalls basic notions about TTS and NTA. Section 3 presents the problem of avoiding shared clocks on examples and rises the problem of comparing NTA component by component. For this, the notion of contextual TTS is developed in Section 4. The problem of avoiding shared clocks is formalized and characterized in terms of contextual TTS. Then Section 5 presents our construction.

This paper details the proofs of the results published in [BC12].

2 Preliminaries

2.1 Timed Transition Systems

The behavior of timed systems is often described as timed transition systems.

Definition 1. A *timed transition system* (TTS) is a tuple $(S, s_0, \Sigma, \rightarrow)$ where S is a set of states, $s_0 \in Q$ is the initial state, Σ is a finite set of actions disjoint from $\mathbb{R}_{\geq 0}$, and $\rightarrow \subseteq S \times (\Sigma \cup \mathbb{R}_{\geq 0}) \times S$ is a set of edges.

For any $a \in \Sigma \cup \mathbb{R}_{\geq 0}$, we write $s \xrightarrow{a} s'$ if $(s, a, s') \in \rightarrow$, and $s \xrightarrow{a}$ if for some s', $(s, a, s') \in \rightarrow$. A *path* of a TTS is a possibly infinite sequence of transitions $\rho = s \xrightarrow{d_0} s'_0 \xrightarrow{a_0} \cdots s_n \xrightarrow{d_n} s'_n \xrightarrow{a_n} \cdots$, where, for all i, $d_i \in \mathbb{R}_{\geq 0}$ and $a_i \in \Sigma$. A path is *initial* if it starts in s_0. A path $\rho = s \xrightarrow{d_0} s'_0 \xrightarrow{a_0} \cdots s_n \xrightarrow{d_n} s'_n \xrightarrow{a_n} s'_n \cdots$ generates a *timed word* $w = (a_0, t_0)(a_1, t_1) \dots (a_n, t_n) \dots$ where, for all i, $t_i = \sum_{k=0}^{i} d_k$. The duration of w is $\delta(w) = \sup_i t_i$ and the untimed word of w is $\lambda(w) = a_0 a_1 \dots a_n \dots$, and we denote the set of timed words over Σ and of duration d as $\mathrm{TW}(\Sigma, d) = \{w \mid \delta(w) = d \wedge \lambda(w) \in \Sigma^*\}$. Lastly, we write $s \xrightarrow{w} s'$ if there is a path from s to s' that generates the timed word w.

In the following definitions, we use two TTS $T_1 = (S_1, s_1^0, \Sigma_1, \rightarrow_1)$ and $T_2 = (S_2, s_2^0, \Sigma_2, \rightarrow_2)$, and $\Sigma_i^{\not\varepsilon}$ denotes $\Sigma_i \setminus \{\varepsilon\}$, where ε is the silent action.

Product of TTS. The *product* of T_1 and T_2, denoted by $T_1 \otimes T_2$, is the TTS $\big(S_1 \times S_2, (s_1^0, s_2^0), \Sigma_1 \cup \Sigma_2, \rightarrow\big)$, where $\rightarrow$ is defined as:

- $(s_1, s_2) \xrightarrow{a} (s'_1, s_2)$ iff $s_1 \xrightarrow{a}_1 s'_1$, for any $a \in \Sigma_1 \setminus \Sigma_2^{\not\varepsilon}$,
- $(s_1, s_2) \xrightarrow{a} (s_1, s'_2)$ iff $s_2 \xrightarrow{a}_2 s'_2$, for any $a \in \Sigma_2 \setminus \Sigma_1^{\not\varepsilon}$,
- $(s_1, s_2) \xrightarrow{a} (s'_1, s'_2)$ iff $s_1 \xrightarrow{a}_1 s'_1$ and $s_2 \xrightarrow{a}_2 s'_2$, for any $a \in (\Sigma_1^{\not\varepsilon} \cap \Sigma_2^{\not\varepsilon}) \cup \mathbb{R}_{\geq 0}$.

Timed Bisimulations. Let $\approx$ be a binary relation over $S_1 \times S_2$. We write $s_1 \approx s_2$ for $(s_1, s_2) \in \approx$. $\approx$ is a *strong timed bisimulation* relation between T_1 and T_2 if $s_1^0 \approx s_2^0$ and $s_1 \approx s_2$ implies that, for any $a \in \Sigma \cup \mathbb{R}_{\geq 0}$, if $s_1 \xrightarrow{a}_1 s_1'$, then, for some s_2', $s_2 \xrightarrow{a}_2 s_2'$ and $s_1' \approx s_2'$; and conversely, if $s_2 \xrightarrow{a}_2 s_2'$, then, for some s_1', $s_1 \xrightarrow{a}_1 s_1'$ and $s_1' \approx s_2'$.

Let $\Rightarrow_i$ (for $i \in \{1, 2\}$) be the transition relation defined as:

- $s \overset{\varepsilon}{\Rightarrow}_i s'$ if $s(\xrightarrow{\varepsilon}_i)^* s'$,
- $\forall a \in \Sigma$, $s \overset{a}{\Rightarrow}_i s'$ if $s(\xrightarrow{\varepsilon}_i)^* \xrightarrow{a}_i (\xrightarrow{\varepsilon}_i)^* s'$,
- $\forall d \in \mathbb{R}_{\geq 0}$, $s \overset{d}{\Rightarrow}_i s'$ if $s(\xrightarrow{\varepsilon}_i)^* \overset{d_0}{\Longrightarrow}_i (\xrightarrow{\varepsilon}_i)^* \cdots \overset{d_n}{\Longrightarrow}_i (\xrightarrow{\varepsilon}_i)^* s'$, where $\sum_{k=0}^{n} d_k = d$.

Then, $\approx$ is a *weak timed bisimulation* relation between T_1 and T_2 if $s_1^0 \approx s_2^0$ and $s_1 \approx s_2$ implies that, for any $a \in \Sigma \cup \mathbb{R}_{\geq 0}$, if $s_1 \xrightarrow{a}_1 s_1'$, then, for some s_2', $s_2 \overset{a}{\Rightarrow}_2 s_2'$ and $s_1' \approx s_2'$; and conversely. We write $T_1 \approx T_2$ (resp. $T_1 \sim T_2$) when there is a strong (resp. weak) timed bisimulation between T_1 and T_2.

2.2 Networks of Timed Automata

The set $\mathcal{B}(X)$ of clock constraints over the set of clocks X is defined by the grammar $g ::= x \bowtie k \mid g \wedge g$, where $x \in X$, $k \in \mathbb{N}$ and $\bowtie \in \{<, \leq, =, \geq, >\}$. Invariants are clock constraints of the form $i ::= x \leq k \mid x < k \mid i \wedge i$.

Definition 2. A *network of timed automata (NTA)* [AD94] is a parallel composition of timed automata (TA) denoted as $A_1 \parallel \cdots \parallel A_n$, with $A_i = (L_i, \ell_i^0, X_i, \Sigma_i, E_i, Inv_i)$ where L_i is a finite set of *locations*, $\ell_i^0 \in L_i$ is the *initial* location, X_i is a finite set of *clocks*, Σ_i is a finite set of *actions*, $E_i \subseteq L_i \times \mathcal{B}(X_i) \times \Sigma_i \times 2^{X_i} \times L_i$ is a set of *edges*, and $Inv_i : L_i \to \mathcal{B}(X_i)$ assigns *invariants* to locations.

If $(\ell, g, a, r, \ell') \in E_i$, we also write $\ell \xrightarrow{g,a,r} \ell'$. For such an edge, g is the *guard*, a the *action* and r the set of clocks to *reset*. $C_i \subseteq X_i$ is the set of clocks reset by A_i and for $i \neq j$, $C_i \cap C_j$ may not be empty.

Semantics. To simplify, we give the semantics of a network of two TA $A_1 \parallel A_2$. We denote by $((\ell_1, \ell_2), v)$ a *state* of the NTA, where ℓ_1 and ℓ_2 are the current locations, and $v : X \to \mathbb{R}_{\geq 0}$, with $X = X_1 \cup X_2$, is a *clock valuation* that maps each clock to its current value. A state is legal only if its valuation v satisfies the invariants of the current locations, denoted by $v \models Inv_1(\ell_1) \wedge Inv_2(\ell_2)$. For each set of clocks $r \subseteq X$, the valuation $v[r]$ is defined by $v[r](x) = 0$ if $x \in r$ and $v[r](x) = v(x)$ otherwise. For each $d \in \mathbb{R}_{\geq 0}$, the valuation $v + d$ is defined by $(v + d)(x) = v(x) + d$ for each $x \in X$. Then, the *TTS generated by* $A_1 \parallel A_2$ is $\mathrm{TTS}(A_1 \parallel A_2) = (S, s_0, \Sigma_1 \cup \Sigma_2, \to)$, where S is the set of legal states, $s_0 = ((\ell_1^0, \ell_2^0), v_0)$, where v_0 maps each clock to 0, and $\to$ is defined by

- Local action: $((\ell_1, \ell_2), v) \xrightarrow{a} ((\ell_1', \ell_2), v')$ iff $a \in \Sigma_1 \setminus \Sigma_2^{\not\varepsilon}$, $\ell_1 \xrightarrow{g,a,r} \ell_1'$, $v \models g$, $v' = v[r]$ and $v' \models Inv_1(\ell_1')$, and similarly for a local action in $\Sigma_2 \setminus \Sigma_1^{\not\varepsilon}$,
- Synchronization: $((\ell_1, \ell_2), v) \xrightarrow{a} ((\ell_1', \ell_2'), v')$ iff $a \neq \varepsilon$, $\ell_1 \xrightarrow{g_1,a,r_1} \ell_1'$, $\ell_2 \xrightarrow{g_2,a,r_2} \ell_2'$, $v \models g_1 \wedge g_2$, $v' = v[r_1 \cup r_2]$ and $v' \models Inv_1(\ell_1') \wedge Inv_2(\ell_2')$,
- Time delay: $\forall d \in \mathbb{R}_{\geq 0}, ((\ell_1, \ell_2), v) \xrightarrow{d} ((\ell_1, \ell_2), v + d)$ iff $\forall d' \in [0, d], v + d' \models Inv_1(\ell_1) \wedge Inv_2(\ell_2)$.

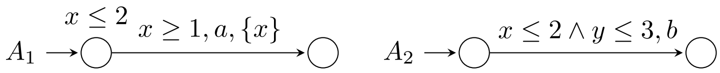

Figure 1: A_2 could avoid reading clock x which belongs to A_1.

A *run* of a NTA is an initial path in its TTS. The semantics of a TA A alone can also be given as a TTS denoted by TTS(A) with only local actions and delay. A TA is *non-Zeno* iff for every infinite timed word w generated by a run, time diverges (i.e. $\delta(w) = \infty$). This is a common assumption for TA. In the sequel, we always assume that the TA we deal with are non-Zeno.

Remark 1. Let $A_1 \parallel A_2$ be such that $X_1 \cap X_2 = \emptyset$. Then $\text{TTS}(A_1) \otimes \text{TTS}(A_2)$ is isomorphic to $\text{TTS}(A_1 \parallel A_2)$. This is not true in general when $X_1 \cap X_2 \neq \emptyset$. For example, in Fig. 2, taking b at time 0.5 and e at time 1 is possible in $\text{TTS}(A_1) \otimes \text{TTS}(A_2)$ but not in $\text{TTS}(A_1 \parallel A_2)$, since b resets x which is tested by e.

3 Need for Shared Clocks

3.1 Problem Setting

We are interested in detecting the cases where it is possible to avoid sharing clocks, so that the model can be implemented using no other synchronization than those explicitly described by common actions.

To start with, let us focus on the case of a network of two TA, $A_1 \parallel A_2$, such that A_1 does not read the clocks reset by A_2, and A_2 may read the clocks reset by A_1. We want to know whether A_2 really needs to read these clocks, or if another NTA $A_1' \parallel A_2'$ could achieve the same behavior as $A_1 \parallel A_2$ without using shared clocks.

A first remark is that our problem makes sense only if we insist on the distributed nature of the system, made of two separate components. On the other hand, if the composition operator is simply used as a convenient syntax for describing a system that is actually implemented on a single sequential component, then a simple product automaton would perfectly describe the system and every clock becomes local.

So, let us consider the example of Fig. 1, made of two TA, supposed to describe two separate components. Remark that A_2 reads clock x which is reset by A_1. But a simple analysis shows that this reading could be avoided: because of the condition on its clock y, A_2 can only take transition b before time 3; but x cannot reach value 2 before time 3, since it must be reset between time 1 and 2. Thus, forgetting the condition on x in A_2 would not change the behavior of the system.

3.2 Transmitting Information during Synchronizations

Consider now the example of Fig. 2. Here also A_2 reads clock x which is reset by A_1, and here also this reading could be avoided. The idea is that A_1 could transmit the value of x when synchronizing, and afterwards, any reading of x in A_2 can be replaced by the reading of a new clock x' dedicated to storing the value of x which is copied on the synchronization. Therefore A_2 can be replaced by A_2' pictured in Fig. 2, while preserving the behavior of the NTA, but also the behavior of A_2 w.r.t. A_1.

We claim that we cannot avoid reading x without this copy of clock. Indeed, after the synchronization, the maximal delay in the current location depends on the exact value of x, and even if we find a mechanism to allow A_2' to move to different locations according to the value of

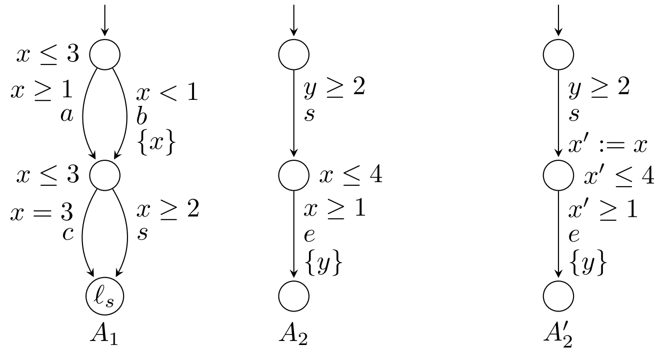

Figure 2: A_2 reads x which belongs to A_1 and A'_2 does not.

x at synchronization time, infinitely many locations would be required (for example, if s occurs at time 2, x may have any value in $(1, 2]$).

Coding Transmission of Information. In order to model the transmission of information during synchronizations, we allow A'_1 and A'_2 to use a larger synchronization alphabet than A_1 and A_2. This allows A'_1 to transmit discrete information like its current location, to A'_2.

But we saw that A'_1 also needs to transmit the exact value of its clocks. For this we allow an automaton to copy its neighbor's clocks into local clocks during synchronizations. This is denoted as updates of the form $x' := x$ in A'_2 (see Fig. 2). This is a special case of updatable timed automata as defined in [BDFP04]. Moreover, as shown in [BDFP04], the class we consider, with diagonal-free constraints and updates with equality (they allow other operators) is not more expressive than classical TA for the sequential semantics (any updatable TA of the class is bisimilar to a classical TA), and the emptiness problem is PSPACE-complete.

Semantics. $\mathrm{TTS}(A_1 \parallel A_2)$ can be defined as previously, with the difference that the synchronizations are now defined by: $((\ell_1, \ell_2), v) \xrightarrow{a} ((\ell'_1, \ell'_2), v')$ iff $\ell_1 \xrightarrow{g_1, a, r_1}_1 \ell'_1$, $\ell_2 \xrightarrow{g_2, a, r_2, u}_2 \ell'_2$ where u is a partial function from X_2 to X_1, $v \models g_1 \wedge g_2$, $v' = (v[r_1 \cup r_2])[u]$, and $v' \models \mathit{Inv}(\ell'_1) \wedge \mathit{Inv}(\ell'_2)$. The valuation $v[u]$ is defined by $v[u](x) = v(u(x))$ if $u(x)$ is defined, and $v[u](x) = v(x)$ otherwise.

Here, we choose to apply the reset $r_1 \cup r_2$ before the update u, because we are interested in sharing the state reached in A_1 after the synchronization, and r_1 may reset some clocks in $C_1 \subseteq X_1$.

3.3 Towards a Formalization of the Problem

We want to know whether A_2 really needs to read the clocks reset by A_1, or if another NTA $A'_1 \parallel A'_2$ could achieve the same behavior as $A_1 \parallel A_2$ without using shared clocks. It remains to formalize what we mean by "having the same behavior" in this context.

First, we impose that the locality of actions is preserved, i.e. A'_1 uses the same set of local actions as A_1, and similarly for A'_2 and A_2. For the synchronizations, we have explained earlier why we allow A'_1 and A'_2 to use a larger synchronization alphabet than A_1 and A_2. The correspondence between the two alphabets will be done by a mapping ψ (this point will be refined later).

Now we have to impose that the behavior is preserved. The first idea that comes in mind is to impose bisimulation between $\psi(\mathrm{TTS}(A'_1 \parallel A'_2))$ (i.e. $\mathrm{TTS}(A'_1 \parallel A'_2)$ with synchronization actions

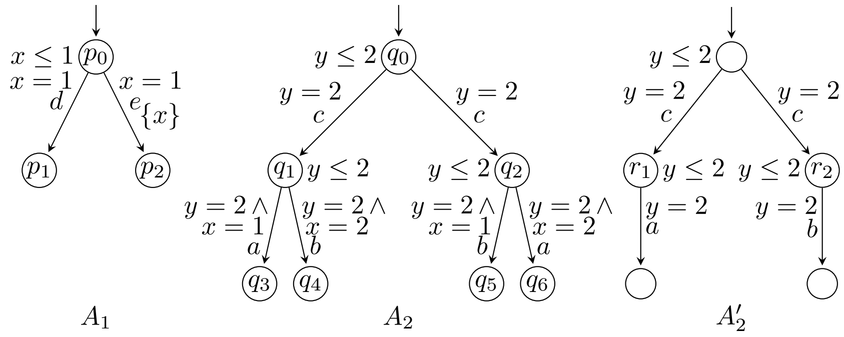

Figure 3: A_2 needs to read the clocks of A_1 and $\mathrm{TTS}(A_1 \parallel A_2) \sim \mathrm{TTS}(A_1 \parallel A'_2)$

relabeled by ψ) and $\mathrm{TTS}(A_1 \parallel A_2)$. But this is not sufficient, as illustrated by the example of Fig. 3 (where ψ is the identity). Intuitively A_2 needs to read x when in q_1 (and similarly in q_2) at time 2, because this reading determines whether it will perform a or b, and the value of x cannot be inferred from its local state given by q_1 and the value of y. Anyway $\mathrm{TTS}(A_1 \parallel A'_2)$ is bisimilar to $\mathrm{TTS}(A_1 \parallel A_2)$, and A'_2 does not read x. For the bisimulation relation $\mathcal{R}$, it suffices to impose $(p_1, q_1)\ \mathcal{R}\ (p_1, r_1)$ and $(p_2, q_1)\ \mathcal{R}\ (p_2, r_2)$.

What we see here is that, if we focus on the point of view of A_2 and A'_2, these two automata do not behave the same. As a matter of fact, when A_2 fires one edge labeled by c, it has not read x yet, and there is still a possibility to fire a or b, whereas when A'_2 fires one edge labeled by c, there is no more choice afterwards. Therefore we need a relation between A'_2 and A_2, and in the general case, a relation between A'_1 and A_1 also.

4 Contextual Timed Transition Systems

As we are interested in representing a partial view of one of the components, we need to introduce another notion, that we call *contextual timed transition system*. This resembles the powerset construction used in game theory to capture the knowledge of an agent about another agent [Rei84].

Notations. $\mathbb{S} = \Sigma_1^{\not\varepsilon} \cap \Sigma_2^{\not\varepsilon}$ denotes the set of common actions. Q_1 denotes the set of states of $\mathrm{TTS}(A_1)$. When $s = ((\ell_1, \ell_2), v)$ is a state of $\mathrm{TTS}(A_1 \parallel A_2)$, we also write $s = (s_1, s_2)$, where $s_1 = (\ell_1, v_{|X_1})$ is in Q_1, and $s_2 = (\ell_2, v_{|X_2 \setminus X_1})$, where $v_{|X}$ is v restricted to X.

Definition 3 (UR(s))**.** Let $\mathrm{TTS}(A_1) = (Q_1, s_0, \Sigma_1, \rightarrow_1)$ and $s \in Q_1$. The set of states of A_1 reachable from s by local actions in 0 delay (and therefore not observable by A_2) is denoted by $\mathrm{UR}(s) = \{s' \in Q_1 \mid \exists w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not\varepsilon}, 0) : s \xrightarrow{w}_1 s'\}$.

Contextual States. The states of this contextual TTS are called *contextual states*. They can be regarded as possibly infinite sets of states of $\mathrm{TTS}(A_1 \parallel A_2)$ for which A_2 is in the same location and has the same valuation over $X_2 \setminus X_1$. A_2 may not be able to distinguish between some states (s_1, s_2) and (s'_1, s_2). In $\mathrm{TTS}_{A_1}(A_2)$, these states are grouped into the same contextual state. However, when $X_2 \cap X_1 \neq \emptyset$, it may happen that A_2 is able to perform a local action or delay from (s_1, s_2) and not from (s'_1, s_2), even if these states are grouped in a same contextual state.

Definition 4 (Contextual TTS)**.** Let $\mathrm{TTS}(A_1 \parallel A_2) = (Q, q_0, \Sigma_1 \cup \Sigma_2, \Rightarrow)$. Then, the *TTS of* A_2 *in the context of* A_1, denoted by $\mathrm{TTS}_{A_1}(A_2)$, is the TTS $(S, s_0, (\Sigma_2 \setminus \mathbb{S}) \cup (\mathbb{S} \times Q_1), \rightarrow)$, where

- $S = \{(S_1, s_2) \mid \forall s_1 \in S_1, (s_1, s_2) \in Q\}$,
- $s_0 = (S_1^0, s_2^0)$, s.t. $(s_1^0, s_2^0) = q_0$ and $S_1^0 = \mathrm{UR}(s_1^0)$,
- $\rightarrow$ is defined by
  - Local action: for any $a \in \Sigma_2 \setminus \mathbb{S}$, $(S_1, s_2) \xrightarrow{a} (S_1', s_2')$ iff $\exists s_1 \in S_1 : (s_1, s_2) \overset{a}{\Rightarrow} (s_1, s_2')$, and $S_1' = \{s_1 \in S_1 \mid (s_1, s_2) \overset{a}{\Rightarrow} (s_1, s_2')\}$
  - Synchronization: for any $(a, s_1') \in \mathbb{S} \times Q_1$, $(S_1, s_2) \xrightarrow{a, s_1'} (\mathrm{UR}(s_1'), s_2')$ iff $\exists s_1 \in S_1 : (s_1, s_2) \overset{a}{\Rightarrow} (s_1', s_2')$
  - Local delay: for any $d \in \mathbb{R}_{\geq 0}$, $(S_1, s_2) \xrightarrow{d} (S_1', s_2')$ iff $\exists s_1 \in S_1, w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not{s}}, d) : (s_1, s_2) \overset{w}{\Rightarrow} (s_1', s_2')$, and $S_1' = \{s_1' \mid \exists s_1 \in S_1, w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not{s}}, d) : (s_1, s_2) \overset{w}{\Rightarrow} (s_1', s_2')\}$

For example, consider A_1 and A_2 of Fig. 3. The initial state is $\big(\{(p_0, 0)\}, (q_0, 0)\big)$. From this contextual state, it is possible to delay 2 time units and reach the contextual state $\big(\{(p_1, 2), (p_2, 1)\}, (q_0, 2)\big)$. Indeed, during this delay, A_1 has to perform either e and reset x, or d. Now, from this contextual state, we can take an edge labeled by c, and reach $\big(\{(p_1, 2), (p_2, 1)\}, (q_1, 2)\big)$. Lastly, from this new state, a can be fired, because it is enabled by $((p_2, 1), (q_1, 2))$ in the TTS of the NTA, and the reached contextual state is $\big(\{(p_2, 1)\}, (q_3, 2)\big)$.

We say that there is no restriction in $\mathrm{TTS}_{A_1}(A_2)$ if whenever a local step is possible from a reachable contextual state, then it is possible from all the states (s_1, s_2) that are grouped into this contextual state. In the example above, there is a restriction in $\mathrm{TTS}_{A_1}(A_2)$ because we have seen that a is enabled only by $((p_2, 1), (q_1, 2))$, and not by all states merged in $\big(\{(p_1, 2), (p_2, 1)\}, (q_1, 2)\big)$. Formally, we use the predicate $\mathit{noRestriction}_{A_1}(A_2)$ defined as follows.

Definition 5 ($\mathit{noRestriction}_{A_1}(A_2)$)**.** The predicate $\mathit{noRestriction}_{A_1}(A_2)$ holds iff for any reachable state (S_1, s_2) of $\mathrm{TTS}_{A_1}(A_2)$, both

- $\forall a \in \Sigma_2 \setminus \mathbb{S}, (S_1, s_2) \xrightarrow{a} (S_1', s_2') \iff \forall s_1 \in S_1, (s_1, s_2) \overset{a}{\Rightarrow} (s_1, s_2')$, and
- $\forall d \in \mathbb{R}_{\geq 0}, (S_1, s_2) \xrightarrow{d} \iff \forall s_1 \in S_1, \exists w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not{s}}, d) : (s_1, s_2) \overset{w}{\Rightarrow}$

Remark 2. If A_2 does not read X_1, then $\mathit{noRestriction}_{A_1}(A_2)$.

Sharing of Information on the Synchronizations. Later we assume that during a synchronization, A_1 is allowed to transmit all its state to A_2, that is why, in $\mathrm{TTS}_{A_1}(A_2)$, we distinguish the states reached after a synchronization according to the state reached in A_1. We also label the synchronization edges by a pair $(a, s_1) \in \mathbb{S} \times Q_1$ where a is the action and s_1 the state reached in A_1.

For the sequel, let $\mathrm{TTS}_{Q_1}(A_1)$ (resp. $\mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$) denote $\mathrm{TTS}(A_1)$ (resp. $\mathrm{TTS}(A_1 \parallel A_2)$) where the synchronization edges are labeled by (a, s_1), where $a \in \mathbb{S}$ is the action, and s_1 is the state reached in A_1.

We can now state a nice property of unrestricted contextual TTS that is similar to the distributivity of TTS over the composition when considering TA with disjoint sets of clocks (see Remark 1). We say that a TA is *deterministic* if it has no ε-transition and for any location ℓ and action a, there is at most one edge labeled by a from ℓ.

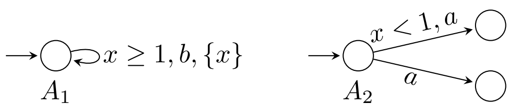

Figure 4: $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2) \approx \mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$, although there is a restriction in $\mathrm{TTS}_{A_1}(A_2)$

Lemma 1. *If there is no restriction in* $\mathrm{TTS}_{A_1}(A_2)$*, then* $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2) \approx \mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$*. Moreover, when A_2 is deterministic, this condition becomes necessary.*

The example of Fig. 4 shows that the reciprocal does not hold when A_2 is not deterministic. In order to prove Lemma 1, we first present two propositions. The first one relates the reachable states of $\mathrm{TTS}_{A_1}(A_2)$ with those of $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$.

Proposition 1.

1. *For any reachable state (S_1, s_2) of* $\mathrm{TTS}_{A_1}(A_2)$*,*
   $s_1 \in S_1 \implies (s_1, (S_1, s_2))$ *is a reachable state of* $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$

2. *noRestriction*$_{A_1}(A_2)$ *iff*
   for any reachable state (S_1, s_2) of $\mathrm{TTS}_{A_1}(A_2)$*,*
   $s_1 \in S_1 \iff (s_1, (S_1, s_2))$ *is a reachable state of* $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$

Proof. (1) For any reachable state (S_1, s_2), let us denote by $P(S_1, s_2)$ the fact that for any $s_1 \in S_1$, $(s_1, (S_1, s_2))$ is reachable in $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$. We give a recursive proof. First, the initial state (S_1^0, s_2^0) satisfies $P(S_1^0, s_2^0)$ because for any $s_1 \in S_1^0 = \mathrm{UR}(s_1^0)$, $\exists w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not\circlearrowleft}, 0) : s_1^0 \xrightarrow{w}_1 s_1$ and hence $(s_1^0, (S_1^0, s_2^0)) \xrightarrow{w} (s_1, (S_1^0, s_2^0))$. Then, assume some reachable state (S_1, s_2) of $\mathrm{TTS}_{A_1}(A_2)$ satisfies $P(S_1, s_2)$ and show that any state (S_1', s_2') reachable in one step from (S_1, s_2) also satisfies $P(S_1', s_2')$. There can be three kinds of steps from (S_1, s_2) in $\mathrm{TTS}_{A_1}(A_2)$.

1. If for some $a \in \Sigma_2 \setminus \mathbb{S}$, $(S_1, s_2) \xrightarrow{a} (S_1', s_2')$, then for any $s_1' \in S_1' \subseteq S_1$, $(s_1', (S_1, s_2)) \xrightarrow{a} (s_1', (S_1', s_2'))$, i.e. $P(S_1', s_2')$ holds.

2. If for some $(a, s_1') \in \mathbb{S} \times Q_1$, $(S_1, s_2) \xrightarrow{a, s_1'} (S_1', s_2')$, then $S_1' = \mathrm{UR}(s_1')$, and for some $s_1 \in S_1$, $(s_1, (S_1, s_2)) \xrightarrow{a, s_1'} (s_1', (S_1', s_2'))$. By the same reasoning as for (S_1^0, s_2^0), for any $s_1'' \in S_1' = \mathrm{UR}(s_1')$, $\exists w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not\circlearrowleft}, 0) : (s_1', (S_1', s_2')) \xrightarrow{w} (s_1'', (S_1', s_2'))$. Hence $P(S_1', s_2')$ holds.

3. If for some $d \in \mathbb{R}_{\geq 0}$, $(S_1, s_2) \xrightarrow{d} (S_1', s_2')$, then $\exists d_1 \leq d : (S_1, s_2) \xrightarrow{d_1} (S_1^1, s_2^1) \wedge \exists s_1^1 \in S_1^1, s_1 \in S_1 : (s_1, s_2) \xRightarrow{d_1} (s_1^1, s_2^1)$, that is $(s_1^1, (S_1^1, s_2^1))$ is reachable, and by time-determinism, $(S_1^1, s_2^1) \xrightarrow{d - d_1} (S_1', s_2')$.

For the third case, take d_1 small enough (but strictly positive) so that $S_1^1 = \{s_1' \mid \exists s_1 \in S_1 : (s_1, s_2) \xRightarrow{d_1} (s_1^1, s_2^1) \wedge s_1' \in \mathrm{UR}(s_1^1)\}$. That is, after some local actions that take no time, A_1 is able to perform a delay d_1 during which no local action is enabled (such d_1 exists because of the non-zenoness assumption). With such d_1, any state $s_1' \in S_1^1$ is such that $s_1' \in \mathrm{UR}(s_1^1)$ for some s_1^1 so that $(s_1^1, (S_1^1, s_2^1))$ is reachable. Therefore, $\exists w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not\circlearrowleft}, 0) : (s_1^1, (S_1^1, s_2^1)) \xrightarrow{w} (s_1', (S_1^1, s_2^1))$ and hence $P(S_1^1, s_2^1)$ holds.

Since A_1 is not Zeno, any delay in $\mathrm{TTS}_{A_1}(A_2)$ can be cut into a finite number of such smaller global delays. Hence, for any (S_1, s_2) that satisfies $P(S_1, s_2)$, for any $d \in \mathbb{R}_{\geq 0}$ such that $(S_1, s_2) \xrightarrow{d} (S'_1, s'_2)$, $P(S'_1, s'_2)$ holds.

(2, $\Rightarrow$) (1) already gives that $\forall s_1 \in S_1$, $(s_1, (S_1, s_2))$ is a reachable state. So it remains to prove that, when $noRestriction_{A_1}(A_2)$, if $(s_1, (S_1, s_2))$ is a reachable state, then $s_1 \in S_1$. We say that a reachable state $s = (s_1, (S_1, s_2))$ satisfies $P(s)$ iff $s_1 \in S_1$.

Assume $noRestriction_{A_1}(A_2)$ and $s = (s_1, (S_1, s_2)$ is a reachable state that satisfies $P(s)$. Then, any state s' reachable in one step from s by some local action or delay $a \in (\Sigma_1 \cup \Sigma_2) \setminus \mathbb{S} \cup \mathbb{R}_{\geq 0}$ or by some synchronization $(a, s'_1) \in \mathbb{S} \times Q_1$ matches one of the following cases.

- if $a \in \Sigma_1 \setminus \Sigma_2^{\not\varnothing}$, then $s' = (s'_1, (S_1, s_2))$ such that $s'_1 \in \mathrm{UR}(s_1) \subseteq S_1$ (by construction, $s_1 \in S_1 \implies \mathrm{UR}(s_1) \subseteq S_1$),
- if $a \in \Sigma_2 \setminus \Sigma_1$, then $s' = (s_1, (S_1, s'_2))$,
- if $a \in \mathbb{R}_{\geq 0}$, then $s' = (s'_1, (S'_1, s'_2))$, where s'_1 such that $(s_1, s_2) \overset{a}{\Rightarrow} (s'_1, s'_2)$ is in $S'_1 = \{q'_1 \mid \exists q_1 \in S_1, w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not\varnothing}, a) : (q_1, s_2) \overset{w}{\Rightarrow} (q'_1, s'_2)\}$,
- if $(a, s'_1) \in (\mathbb{S} \times Q_1)$, then $s' = (s'_1, (\mathrm{UR}(s'_1), s'_2))$.

Therefore, any state s' reached in one step from s also satisfies $P(s')$, and recursively, since the initial state $s_0 = (s_1^0, (\mathrm{UR}(s_1^0), s_2^0))$ satisfies $P(s_0)$, any reachable state s of $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$ satisfies $P(s)$.

(2, $\Leftarrow$) By contradiction, assume there is a restriction in state (S_1, s_2) for local delay or action $a \in (\Sigma_2 \setminus \Sigma_1) \cup \mathbb{R}_{\geq 0}$ i.e. a is possible from some state (s'_1, s_2) but not from another state (s_1, s_2) such that $s'_1, s_1 \in S_1$. Then, after performing a from $(s_1, (S_1, s_2))$, that is reachable according to Proposition 1, we reach state $(s_1, (S'_1, s'_2))$ such that $s_1 \notin S'_1$. $\square$

Proposition 2. *If $noRestriction_{A_1}(A_2)$ then, for all timed word w over $(\Sigma_2 \setminus \mathbb{S}) \cup (\mathbb{S} \times Q_1)$, there exists at most one S_1 such that, for some s_2, $(S_1^0, s_2^0) \xrightarrow{w} (S_1, s_2)$ in $\mathrm{TTS}_{A_1}(A_2)$ (i.e. S_1 is uniquely determined by w, whatever the structure of A_2).*

Proof. Assume $noRestriction_{A_1}(A_2)$, we show that, for any (S_1^1, s_2^1) reachable in $\mathrm{TTS}_{A_1}(A_2)$, for any action or delay in $(\Sigma_2 \setminus \mathbb{S}) \cup (\mathbb{S} \times Q_1) \cup \mathbb{R}_{\geq 0}$, there is at most one S_1 such that, for some s_2, (S_1, s_2) is a successor of (S_1^1, s_2^1) by this action.

Indeed, by construction, and since there is no restriction,

- any successor of (S_1^1, s_2^1) by a local action is of the form (S_1^1, s'_2),
- any successor of (S_1^1, s_2^1) by a synchronization (a, s'_1) is of the form $(\mathrm{UR}(s'_1), s'_2)$,
- any successor of (S_1^1, s_2^1) by a delay d is of the form (S_1, s'_2) with $S_1 = \{s'_1 \mid \exists w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not\varnothing}, d), s_1 \in S_1^1 : s_1 \xrightarrow{w}_1 s'_1\}$.

Therefore, for any possible action or delay, S_1 does not depend on the state of A_2, and is uniquely determined by this action or delay.

Since (S_1^0, s_2^0) is unique, for any timed word w over $(\Sigma_2 \setminus \mathbb{S}) \cup (\mathbb{S} \times Q_1)$, either w does not describe a valid path in $\mathrm{TTS}_{A_1}(A_2)$, or there exists a unique S_1 such that for some s_2, $(S_1^0, s_2^0) \xrightarrow{w} (S_1, s_2)$ in $\mathrm{TTS}_{A_1}(A_2)$. $\square$

We can now prove Lemma 1.

Proof of Lemma 1. Assume $noRestriction_{A_1}(A_2)$, and define relation $\mathcal{R}$ as $(s_1, (S_1, s_2))\ \mathcal{R}\ (s'_1, s'_2) \overset{def}{\iff} s_1 = s'_1 \wedge s_2 = s'_2$, for any reachable states $(s_1, (S_1, s_2))$ of $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$ and (s'_1, s'_2) of $\mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$. By Proposition 1, since $(s_1, (S_1, s_2))$ is reachable, $s_1 \in S_1$. We show that $\mathcal{R}$ is a strong timed bisimulation.

First, the initial states are $\mathcal{R}$-related: $(s^0_1, (S^0_1, s^0_2))\ \mathcal{R}\ (s^0_1, s^0_2)$. Then, if $(s_1, (S_1, s_2))\ \mathcal{R}\ (s'_1, s'_2)$, four kinds of steps are possible:

- if for some $a \in \Sigma_1 \setminus \Sigma_2^{\not{S}}$, $(s_1, (S_1, s_2)) \xrightarrow{a} (s'_1, (S_1, s_2))$, then $(s_1, s_2) \xRightarrow{a} (s'_1, s_2)$ and $(s'_1, (S_1, s_2))\ \mathcal{R}\ (s'_1, s_2)$, and conversely.
- if for some $a \in \Sigma_2 \setminus \Sigma_1$, $(s_1, (S_1, s_2)) \xrightarrow{a} (s_1, (S_1, s'_2))$, then, $\forall s_{11} \in S_1$, $(s_{11}, s_2) \xRightarrow{a} (s_{11}, s'_2)$ (because $noRestriction_{A_1}(A_2)$), and in particular, $(s_1, s_2) \xRightarrow{a} (s_1, s'_2)$ and $(s_1, (S_1, s'_2))\ \mathcal{R}\ (s_1, s'_2)$, and conversely.
- if for some $(a, s'_1) \in \mathbb{S} \times Q_1$, $(s_1, (S_1, s_2)) \xrightarrow{a, s'_1} (s'_1, (S'_1, s'_2))$, then $(s_1, s_2) \xRightarrow{a, s'_1} (s'_1, s'_2)$ and $(s'_1, (S'_1, s'_2))\ \mathcal{R}\ (s'_1, s'_2)$, and conversely.
- if for some $d \in \mathbb{R}_{\geq 0}$, $(s_1, (S_1, s_2)) \xrightarrow{d} (s'_1, (S'_1, s'_2))$, then $(s_1, s_2) \xRightarrow{d} (s'_1, s'_2)$ (because $noRestriction_{A_1}(A_2)$), and $(s'_1, (S'_1, s'_2))\ \mathcal{R}\ (s'_1, s'_2)$, and conversely.

Now assume A_2 is deterministic. Let relation $\mathcal{R}$ be a strong timed bisimulation between $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$ and $\mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$.

By contradiction, assume there is a restriction in $\mathrm{TTS}_{A_1}(A_2)$. Then there is a reachable state (S_1, s_2) of $\mathrm{TTS}_{A_1}(A_2)$, and a local delay or action $a \in (\Sigma_2 \setminus \Sigma_1) \cup \mathbb{R}_{\geq 0}$ such that, for some $s_1, s'_1 \in S_1$, (s_1, s_2) enables a in $\mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$, whereas (s'_1, s_2) does not.

By definition of a bisimulation, there also exist two states $(p_1, (P_1, p_2))$ and $(p'_1, (P'_1, p'_2)$ such that $(p_1, (P_1, p_2))\ \mathcal{R}\ (s_1, s_2)$ and $(p'_1, (P'_1, p'_2))\ \mathcal{R}\ (s'_1, s_2)$. That is, in particular, $(p'_1, (P'_1, p'_2))$ does not enable a. Moreover, these states can be chosen so that they are reached by the same timed word over $(\Sigma_2 \setminus \mathbb{S}) \cup (\mathbb{S} \times Q_1)$, and since A_2 is deterministic, $p_2 = p'_2 = s_2$.

Now, we can assume that (S_1, s_2) is chosen so that it is the first state with a restriction along an initial path. Then, the paths to (P_1, s_2) and (P'_1, s_2) generate the same timed word over $(\Sigma_2 \setminus \mathbb{S}) \cup (\mathbb{S} \times Q_1)$, and by Proposition 2, $P_1 = P'_1 = S_1$.

Therefore, we have shown the existence of a state $(p'_1, (S_1, s_2))$ in $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$ that does not enable a, which means that (S_1, s_2) does not enable a in $\mathrm{TTS}_{A_1}(A_2)$. This contradicts the fact that there exists $s_1 \in S_1$ such that (s_1, s_2) enables a. $\square$

4.1 Need for Shared Clocks Revisited

We have argued in Section 3.3 that the existence of a NTA $A'_1 \parallel A'_2$ without shared clocks and such that $\psi(\mathrm{TTS}_{Q'_1}(A'_1 \parallel A'_2)) \sim \mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$ is not sufficient to capture the idea that A_2 does not need to read the clocks of A_1. We are now equipped to define the relations we want to impose on the separate components, namely $\psi(\mathrm{TTS}_{Q'_1}(A'_1)) \sim \mathrm{TTS}_{Q_1}(A_1)$ and $\psi(\mathrm{TTS}_{A'_1}(A'_2)) \sim \mathrm{TTS}_{A_1}(A_2)$. And since we have seen the importance of using labeling the synchronization actions in contextual TTS by labels in $\mathbb{S} \times Q_1$ rather than in $\mathbb{S}$, the correspondence between the synchronization labels of $A'_1 \parallel A'_2$ with those of $A_1 \parallel A_2$ is now done by a mapping $\psi : \mathbb{S}' \times Q'_1 \to \mathbb{S} \times Q_1$.

This settles the problem of the example of Fig. 3 where $\mathrm{TTS}_{A_1}(A'_2) \not\sim \mathrm{TTS}_{A_1}(A_2)$ (here $A'_1 = A_1$), but as shown in Fig. 5, a problem remains. In this example, we can see that A_2 needs to read clock x of A_1 to know whether it has to perform a or b at time 2, and yet $\mathrm{TTS}_{A_1}(A_2) \sim \mathrm{TTS}_{A_1}(A'_2)$ (here also $A'_1 = A_1$). The intuition to understand this is that the

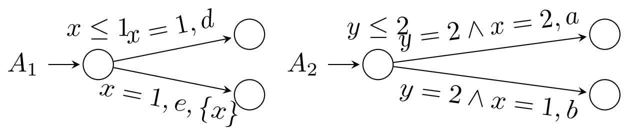

Figure 5: A_2 needs to read the clocks of A_1 and $\mathrm{TTS}_{A_1}(A_2) \sim \mathrm{TTS}_{A_1}(A'_2)$.

contextual TTS merge too many states for the two systems to remain differentiable. However we remark that here, the first condition that we have required in Section 3, namely the global bisimulation between $\psi(\mathrm{TTS}(A'_1 \parallel A'_2))$ and $\mathrm{TTS}(A_1 \parallel A_2)$, does not hold.

Now we show that the conjunction of global and local bisimulations actually gives the good definition.

Definition 6 (Need for shared clocks)**.** Given $A_1 \parallel A_2$ such that A_1 does not read the clocks of A_2, A_2 *does not need to read the clocks of* A_1 iff there exists a NTA $A'_1 \parallel A'_2$ without shared clocks (but with clock copies during synchronizations), using the same sets of local actions and a synchronization alphabet $\mathbb{S}'$ related to the original one by a mapping $\psi : \mathbb{S}' \times Q'_1 \to \mathbb{S} \times Q_1$, and such that

1. $\psi(\mathrm{TTS}_{Q'_1}(A'_1 \parallel A'_2)) \sim \mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$ and
2. $\psi(\mathrm{TTS}_{Q'_1}(A'_1)) \sim \mathrm{TTS}_{Q_1}(A_1)$ and
3. $\psi(\mathrm{TTS}_{A'_1}(A'_2)) \sim \mathrm{TTS}_{A_1}(A_2)$.

Notice that this does not mean that the clock constraints that read X_1 can simply be removed from A_2 (see Fig. 2).

Lemma 2. *When noRestriction*$_{A_1}(A_2)$ *holds, any NTA* $A'_1 \parallel A'_2$ *without shared clocks and that satisfies items 2 and 3 of Definition 6 also satisfies item 1.*

Proof. When *noRestriction*$_{A_1}(A_2)$ holds, then by Lemma 1, $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2) \approx \mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$. So for any NTA $A'_1 \parallel A'_2$ satisfying items 2 and 3 of Definition 6, we have $\psi(\mathrm{TTS}_{Q'_1}(A'_1)) \otimes \psi(\mathrm{TTS}_{A'_1}(A'_2)) \sim \mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$. It remains to show that $\psi(\mathrm{TTS}_{Q'_1}(A'_1 \parallel A'_2)) \approx \psi(\mathrm{TTS}_{Q'_1}(A'_1)) \otimes \psi(\mathrm{TTS}_{A'_1}(A'_2))$. Remark that applying ψ to the labels before doing the product, allows more synchronizations than applying ψ on the TTS of the system since ψ may merge different labels. We show that, in our case, the two resulting TTS are bisimilar anyway.

For this, let $\mathcal{R}_1$ be a bisimulation relation between $\psi(\mathrm{TTS}_{Q'_1}(A'_1))$ and $\mathrm{TTS}_{Q_1}(A_1)$, and $\mathcal{R}_2$ be a bisimulation relation between $\psi(\mathrm{TTS}_{A'_1}(A'_2))$ and $\mathrm{TTS}_{A_1}(A_2)$. We will build inductively a bisimulation $\mathcal{R}$ between $\psi(\mathrm{TTS}_{Q'_1}(A'_1 \parallel A'_2))$ and $\psi(\mathrm{TTS}_{Q'_1}(A'_1)) \otimes \psi(\mathrm{TTS}_{A'_1}(A'_2))$ such that for any (q_1, q_2) and (r_1, r_2) such that $(q_1, q_2)\ \mathcal{R}\ (r_1, r_2)$, there exists a state s_1 of $\mathrm{TTS}_{Q_1}(A_1)$ and a state s_2 of $\mathrm{TTS}_{A_1}(A_2)$ such that $q_1\ \mathcal{R}_1\ s_1$ and $r_1\ \mathcal{R}_1\ s_1$ and $q_2\ \mathcal{R}_2\ s_2$ and $r_2\ \mathcal{R}_2\ s_2$. The inductive definition of $\mathcal{R}$ is as follows. The initial states (which are the same in both sides) are in relation; $\mathcal{R}$ is preserved by delays; $\mathcal{R}$ is preserved by playing local actions. The key is the treatment of synchronizations: when $(q_1, q_2)\ \mathcal{R}\ (r_1, r_2)$ and $q_1 \xrightarrow{a_1} q'_1$ in $\mathrm{TTS}_{Q_1}(A_1)$ and $q_2 \xrightarrow{a_2} q'_2$ in $\mathrm{TTS}_{A_1}(A_2)$ with $\psi(a_1) = \psi(a_2) = a$, then the existence of the s_1 and s_2 mentioned earlier ensures that there exists a state (r'_1, r'_2) in $\psi(\mathrm{TTS}_{Q'_1}(A'_1 \parallel A'_2))$ such that $(r_1, r_2) \xrightarrow{a} (r'_1, r'_2)$, and we set $(q'_1, q'_2)\ \mathcal{R}\ (r'_1, r'_2)$ for any such (r'_1, r'_2).

$\square$

We are now ready to give a criterion to decide the need for shared clocks.

Theorem 1. *When noRestriction$_{A_1}(A_2)$ holds, A_2 does not need to read the clocks of A_1. When A_2 is deterministic, this condition becomes necessary.*

Proof of Theorem 1, necessary condition when A_2 is deterministic. Like in the proof of Lemma 2, we show that for any NTA $A'_1 \parallel A'_2$ satisfying items 2 and 3 of Definition 6, $\psi(\mathrm{TTS}_{Q'_1}(A'_1 \parallel A'_2)) \sim \mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$. But, by Lemma 1, when A_2 is deterministic and $\mathrm{TTS}_{A_1}(A_2)$ has restrictions, $\mathrm{TTS}_{Q_1}(A_1) \otimes \mathrm{TTS}_{A_1}(A_2)$ is not timed bisimilar to $\mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$ (not even weakly timed bisimilar since there are no ε-transitions). Hence any NTA $A'_1 \parallel A'_2$ satisfying items 2 and 3 of Definition 6, does not satisfy item 1. □

We remark from the proof that when there is a restriction in $\mathrm{TTS}_{A_1}(A_2)$, even infinite A'_1 and A'_2 would not help. Next section will be devoted to the constructive proof of the direct part of this theorem.

The counterexample in Fig. 4 also works here to argue that the conditions of Lemma 2 and Theorem 1 are not necessary when A_2 is not deterministic. Indeed A'_2 with only one unguarded edge labeled by a and $A'_1 = A_1$ satisfy the three items of Definition 6 but there is a restriction in $\mathrm{TTS}_{A_1}(A_2)$.

5 Constructing a NTA without Shared Clocks

This section is dedicated to proving Theorem 1 by constructing suitable A'_1 and A'_2. To simplify, we assume that in A_2, the guards on the synchronizations do not read X_1.

5.1 Construction

First, our A'_1 is obtained from A_1 by replacing all the labels $a \in \mathbb{S}$ on the synchronization edges of A_1 by $(a, \ell_1) \in \mathbb{S} \times L_1$, where ℓ_1 is the output location of the edge. Therefore the synchronization alphabet between A'_1 and A'_2 will be $\mathbb{S}' = \mathbb{S} \times L_1$, which allows A'_1 to transmit its location after each synchronization.

Then, the idea is to build A'_2 as a product $A_{1,2} \otimes A_{2,mod}$ ($\otimes$ denotes the product of TA as it is usually defined [AD94]), where $A_{2,mod}$ plays the role of A_2 and $A_{1,2}$ acts as a local copy of A'_1, from which $A_{2,mod}$ reads clocks instead of reading those of A'_1. For this, as long as the automata do not synchronize, $A_{1,2}$ will evolve, simulating a run of A'_1 that is compatible with what A'_2 knows about A'_1. And, as soon as A'_1 synchronizes with A'_2, A'_2 updates $A_{1,2}$ to the actual state of A'_1. If the clocks of $A_{1,2}$ always give the same truth value to the guards and invariants of $A_{2,mod}$ than the actual value of the clocks of A'_1, then our construction behaves like $A_1 \parallel A_2$. To check that this is the case, we equip A'_2 with an error location, ☹, and edges that lead to it if there is a contradiction between the values of the clocks of A'_1 and the values of the clocks of $A_{1,2}$. The guards of these edges are the only cases where A'_2 reads clocks of A'_1. Therefore, if ☹ is not reachable, they can be removed so that A'_2 does not read the clocks of A'_1. More precisely, a contradiction happens when $A_{2,mod}$ is in a given location and the guard of an outgoing edge is true according to $A_{1,2}$ and false according to A'_1, or vice versa, or when the invariant of the current location is false according to A'_1 (whereas it is true according to $A_{1,2}$, since $A_{2,mod}$ reads the clocks of $A_{1,2}$).

Namely, $\mathcal{S}_{mod} = A'_1 \parallel (A_{1,2} \otimes A_{2,mod})$ where $A_{1,2}$ and $A_{2,mod}$ are defined as follows. $A_{1,2} = (L_1, \ell_1^0, X'_1, \mathbb{S}' \cup \{\varepsilon\}, E'_1, \mathit{Inv}'_1)$, where

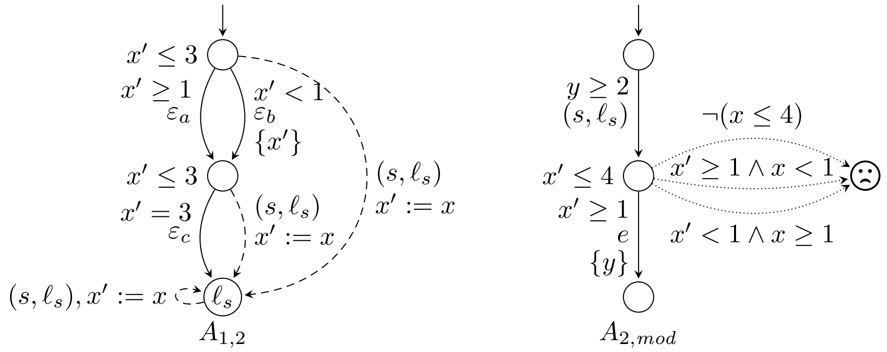

Figure 6: $A_{1,2}$ and $A_{2,mod}$ for the example of Fig. 2

- each clock $x' \in X_1'$ is associated with a clock $c(x') = x \in X_1$ (c is a bijection from X_1' to X_1). For any clock constraint γ, γ' denotes the clock constraint where any clock x of X_1 is substituted by x' of X_1'.
- $\forall \ell \in L_1, Inv_1'(\ell) = Inv_1(\ell)'$
- $E_1' = \{\ell_1 \xrightarrow{g', \varepsilon_a, r'} \ell_2 \mid \exists a \in \Sigma_1 \setminus \Sigma_2^{\not\varepsilon} : \ell_1 \xrightarrow{g, a, c(r')} \ell_2 \in E_1\}$
  $\quad \cup \; \{\ell \xrightarrow{\top, (a, \ell_2), c} \ell_2 \mid \ell \in L_1 \wedge a \in \mathbb{S} \wedge \exists \ell_1 \xrightarrow{g, a, r} \ell_2 \in E_1\}$
  where $\top$ means true, and c denotes the assignment of any clock $x' \in X_1'$ with the value of its associated clock $c(x') = x \in X_1$ (written $x' := x$ in Fig. 6).

$A_{2,mod} = (L_2 \cup \{☹\}, \ell_2^0, X_2 \cup X_1', (\Sigma_2 \setminus \Sigma_1) \cup \mathbb{S}', E_2', Inv_2')$, where

- $\forall \ell \in L_2, Inv_2'(\ell) = Inv_2(\ell)'$ and $Inv_2'(☹) = \top$,
- $E_2' = \{\ell_1 \xrightarrow{g', a, r} \ell_2 \mid \ell_1 \xrightarrow{g, a, r} \ell_2 \in E_2 \wedge a \notin \mathbb{S}\}$
  $\quad \cup \; \{\ell_1 \xrightarrow{g, (a, \ell), r} \ell_2 \mid \ell_1 \xrightarrow{g, a, r} \ell_2 \in E_2 \wedge a \in \mathbb{S} \wedge \ell \in L_1\}$
  $\quad \cup \; \{\ell \xrightarrow{\neg Inv_2(\ell), \varepsilon, \emptyset} ☹ \mid \ell \in L_2\}$
  $\quad \cup \; \{\ell \xrightarrow{g' \wedge \neg g, \varepsilon, \emptyset} ☹ \mid \ell \xrightarrow{g, a, r} \ell' \in E_2 \wedge a \notin \mathbb{S}\}$
  $\quad \cup \; \{\ell \xrightarrow{\neg g' \wedge g, \varepsilon, \emptyset} ☹ \mid \ell \xrightarrow{g, a, r} \ell' \in E_2 \wedge a \notin \mathbb{S}\}$.

For the example of Fig. 2, $A_{1,2}$ and $A_{2,mod}$ are pictured in Fig. 6.

We now prove the correspondence between a state of $\mathcal{S}_{mod}$ and two states of $\text{TTS}(A_1 \parallel A_2)$ that are merged into the same state of $\text{TTS}_{A_1}(A_2)$. This is stated in the following proposition. A state of $\mathcal{S}_{mod}$ is denoted as $(s_1, s_{1,2}, s_2) = \big((\ell_1, v_{|X_1}), (\ell_{1,2}, v_{|X_1'}), (\ell_2, v_{|X_2 \setminus X_1})\big)$. For a given state of $A_{1,2}$, $s_{1,2} = (\ell_{1,2}, v_{|X_1'})$, we denote by $s_{1,2}'$ the state $(\ell_{1,2}, v')$, where $v' : X_1 \to \mathbb{R}_{\geq 0}$ is defined as: for any $x \in X_1$, $v'(x) = v(x')$ (i.e. $s_{1,2}'$ is a state of A_1). Reciprocally, for a given state of A_1, $s_{1,2}' = (\ell_{1,2}, v')$, $s_{1,2}$ denotes the state $(\ell_{1,2}, v)$, where $v : X_1' \to \mathbb{R}_{\geq 0}$ is defined as: for any $x' \in X_1'$, $v(x') = v'(x)$.

Proposition 3. *Let $(s_1, s_{1,2}, s_2)$ be a state of $\mathcal{S}_{mod}$. If along one path that leads to $(s_1, s_{1,2}, s_2)$ no edge leading to ☹ is enabled, then there exists S_1 such that (S_1, s_2) is a reachable state of $\text{TTS}_{A_1}(A_2)$ and s_1 and $s_{1,2}'$ are both in S_1.*

Conversely, let (S_1, s_2) be a reachable state of $\text{TTS}_{A_1}(A_2)$, and s_1 and $s_{1,2}'$ be some states in S_1. Then $(s_1, s_{1,2}, s_2)$ is a state of $\mathcal{S}_{mod}$.

Proof of Proposition 3. Let $(s_1, s_{1,2}, s_2)$ be a reachable state of $\mathcal{S}_{mod}$, such that there is a path ρ from the initial state $(s_1^0, s_{1,2}^0, s_2^0)$ to $(s_1, s_{1,2}, s_2)$ that does not enable any edges leading to ☹ (except maybe from $(s_1, s_{1,2}, s_2)$). We give a recursive proof. First, for the initial state $(s_1^0, s_{1,2}^0, s_2^0)$ of $\mathcal{S}_{mod}$, s_1^0 and $s_{1,2}^{0\prime}$ are bot in S_1^0 such that (S_1^0, s_2^0) is the initial state of $\mathrm{TTS}_{A_1}(A_2)$. Now, assume this is true for some $(p_1, p_{1,2}, p_2)$ visited along ρ. That is, there exists P_1 such that (P_1, p_2) is reachable and $p_1, p'_{1,2} \in P_1$. Then, the next state s' visited along ρ is reached after one of the following steps:

- local action in A'_1: $s' = (q_1, p_{1,2}, p_2)$ such that $q_1 \in \mathrm{UR}(p_1) \subseteq P_1$,
- local action in $A_{1,2}$: $s' = (p_1, q_{1,2}, p_2)$ such that $q'_{1,2} \in \mathrm{UR}(p'_{1,2}) \subseteq P_1$,
- local action in A_2: $s' = (p_1, p_{1,2}, q_2)$ such that there exists S'_1 such that (S'_1, q_2) is reachable from (P_1, q_2) by the same action, and, since no edge leading to ☹ is enabled, both (p_1, p_2) and $(p'_{1,2}, p_2)$ enable this step in $\mathrm{TTS}(A_1 \parallel A_2)$. Therefore, $p_1, p'_{1,2} \in S'_1$.
- synchronization: $s' = (q_1, q_{1,2}, q_2)$ such that there exists $S'_1 = \mathrm{UR}(q_1)$ such that (S'_1, q_2) is reachable from (P_1, q_2) by the same action, and $q_1 = q'_{1,2} \in S'_1$.

By recursion, $(s_1, s_{1,2}, s_2)$ also satisfies the property, that is, there exists S_1 such that (S_1, s_2) is reachable and $s_1, s'_{1,2} \in S_1$.

Conversely, let denote by $P(S_1, s_2)$ the fact that for any reachable state (S_1, s_2) of $\mathrm{TTS}_{A_1}(A_2)$, for any states $s_1, s'_{1,2} \in S_1$, $(s_1, s_{1,2}, s_2)$ is a reachable state of $\mathcal{S}_{mod}$. First, for any $s_1, s'_{1,2} \in S_1^0 = \mathrm{UR}(s_1^0)$, $(s_1, s_{1,2}, s_2^0)$ is a reachable state, because by construction, $A_{1,2}$ can only mimic (as long as there is no synchronization) one possible behavior of A_1 to reach $s_{1,2}$ from s_1^0, therefore $P(S_1^0, s_2^0)$ holds. Assume that for some reachable state (S_1, s_2) $P(S_1, s_2)$ holds. Then any state reachable in one step from (S_1, s_2) is reached by one of the following steps.

- If for some $a \in \Sigma_2 \setminus \mathbb{S}$, $(S_1, s_2) \xrightarrow{a} (S'_1, s'_2)$, then for any $s_1, s'_{1,2} \in S'_1 \subseteq S_1$, $(s_1, s'_{1,2}, s_2) \xrightarrow{a} (s_1, s'_{1,2}, s'_2)$, i.e. $P(S'_1, s'_2)$ holds.
- If for some $(a, s'_1) \in \mathbb{S} \times Q_1$, $(S_1, s_2) \xrightarrow{a, s'_1} (S'_1, s'_2)$, then $S'_1 = \mathrm{UR}(s'_1)$, and for any $s_1, s'_{1,2} \in S'_1$, $(s_1, s_{1,2}, s'_2)$ can be reached from some $(p_1, p_{1,2}, s_2)$ such that $p_1, p'_{1,2} \in S_1$. Indeed, in $\mathcal{S}_{mod}$, synchronization $((a, \ell'_1), s'_1)$ resets $A_{1,2}$ in the same state as A_1 and then A_1 performs some local actions while $A_{1,2}$ also performs some local actions mimicking one possible behavior of A_1 (that is why $s'_{1,2} \in S'_1$). Hence $P(S'_1, s'_2)$ holds.
- If for some $d \in \mathbb{R}_{\geq 0}$, $(S_1, s_2) \xrightarrow{d} (S'_1, s'_2)$, then we use the same reasoning as for a synchronization. Since $A_{1,2}$ is built so that it mimics any possible behavior of A_1 between synchronizations, any state $s'_{1,2} \in S'_1$ reachable by A_1 during this delay corresponds to a state $s_{1,2}$ reachable by $A_{1,2}$. Hence $P(S'_1, s'_2)$ also holds.

By recursion, $P(S_1, s_2)$ holds for any reachable state (S_1, s_2). □

Lastly, the following lemma will be used to prove the direct part of Theorem 1.

Lemma 3. *☹ is reachable in $\mathcal{S}_{mod}$ iff there is a restriction in* $\mathrm{TTS}_{A_1}(A_2)$.

Proof. Assume ☹ is not reachable in $\mathcal{S}_{mod}$. From Proposition 3, we know that for any state (S_1, s_2) of $\mathrm{TTS}_{A_1}(A_2)$, for any s_1, $s'_{1,2}$ in S_1, there is a corresponding state $s = \big((\ell_1, v_{|X_1}), (\ell_{1,2}, v_{|X'_1}), (\ell_2, v_{|X_2 \setminus X_1})\big) = (s_1, s_{1,2}, s_2)$ of $\mathcal{S}_{mod}$. Moreover, for any such s, if there is an outgoing edge towards ☹ from ℓ_2, then this edge is never enabled. That is, for any time constraint γ read in ℓ_2 in the original system $\mathcal{S}$ (invariant of ℓ_2 or guard of an outgoing edge with a local action), $v_{|X_2 \cup X_1} \models \gamma \iff v_{|(X_2 \setminus X_1) \cup X_1'} \models \gamma'$. Hence for any enabled step from (S_1, s_2), s_1 and $s'_{1,2}$ are in the same restriction. Therefore, $noRestriction_{A_1}(A_2)$.

Assume ☹ is reachable in $\mathcal{S}_{mod}$. From Proposition 3, we know that for any state $s = \big((\ell_1, v_{|X_1}), (\ell_{1,2}, v_{|X_1'}), (\ell_2, v_{|X_2 \setminus X_1})\big) = (s_1, s_{1,2}, s_2)$ of $\mathcal{S}_{mod}$, reached after a path that does not enable edges leading to ☹ (except maybe from this last state), there is a corresponding state (S_1, s_2) of $\mathrm{TTS}_{A_1}(A_2)$ such that s_1 and $s'_{1,2}$ are both in S_1. If ☹ can be reached, then consider a path that reach ☹ and such no edge leading to ☹ was enabled before along the path. The last state s of $\mathcal{S}_{mod}$ visited before ☹ is such that for some time constraint γ evaluated at s from ℓ_2, $v_{|X_2 \cup X_1} \models \gamma$ and $v_{|(X_2 \setminus X_1) \cup X_1'} \not\models \gamma'$ (or conversely). Therefore, a local action or local delay is possible from (s_1, s_2) and not from $(s'_{1,2}, s_2)$. Hence (S_1, s_2) is a state with a restriction. □

We now give a first simple case for which Theorem 1 can be proved easily. We say that A_1 has no urgent synchronization if for any location, when the invariant reaches its limit, a local action is enabled. Under this assumption, we can show that $A_2' = A_{1,2} \otimes A'_{2,mod}$, where $A'_{2,mod}$ is $A_{2,mod}$ without location ☹ (that is never reached according to Lemma 3) and its ingoing edges, is suitable. Indeed, we can show that A_2' does not read X_1 and is such that $\psi(\mathrm{TTS}_{A_1'}(A_2')) \sim \mathrm{TTS}_{A_1}(A_2)$, where for any $((a, \ell_1), s_1) \in \mathbb{S}' \times Q_1'$, $\psi(((a, \ell_1), s_1)) = (a, s_1)$. Obviously, item 2 of Definition 6 holds, and Lemma 2 says that item 1 also holds.

When A_1 has urgent synchronizations, this construction allows one to check the absence of restriction in $\mathrm{TTS}_{A_1}(A_2)$, but it does not give directly a suitable A_2'. We will give the idea of the construction of A_2' for the general case later.

Proof of Theorem 1, direct part, when no urgent synchronization in A_1. Assume $noRestriction_{A_1}(A_2)$. We consider $A_2' = A_{1,2} \otimes A'_{2,mod}$ where $A'_{2,mod}$ is $A_{2,mod}$ without ☹ (that is never reached according to Lemma 3) and its ingoing edges. Therefore, $A'_{2,mod}$ does not read X_1 and neither does $A_2' = A_{1,2} \otimes A'_{2,mod}$. Below we show that A_2' is a suitable candidate because $\psi(\mathrm{TTS}_{A_1'}(A_2')) \sim \mathrm{TTS}_{A_1}(A_2)$ ($\psi(\mathrm{TTS}_{Q_1'}(A_2')) \sim \mathrm{TTS}_{Q_1}(A_1)$ obviously holds).

Let $\mathcal{R}$ be the relation such that for any reachable state (S_1, s_2) of $\mathrm{TTS}_{A_1}(A_2)$, and any reachable state (S_1', s_2') of $\psi(\mathrm{TTS}_{A_1'}(A_2'))$,

$$(S_1, s_2)\ \mathcal{R}\ (S_1', s_2') \overset{def}{\iff} \begin{cases} s_2 = (\ell_2, v_2) \text{ and } s_2' = ((\ell_{1,2}, \ell_2), v_2') \text{ s.t.} \\ \forall x \in X_2 \setminus X_1, v_2(x) = v_2'(x) \\ S_1 = S_1' \end{cases}$$

i.e. A_2 and $A'_{2,mod}$ are both in ℓ_2 and their local clocks have the same value, and A_1 and A_1' are in indistinguishable states (states merged in a same contextual state S_1). Obviously, the initial states, (S_1^0, s_2^0) and $(S_1^0, s_2^{0\prime})$, are $\mathcal{R}$-related. Since there is no marked state in $\mathrm{TTS}_{A_1}(A_2)$ (resp. in $\mathrm{TTS}_{A_1'}(A_2')$), for any state $s = (S_1, s_2)$ (resp. $s' = (S_1', s_2')$) of this TTS, all time constraints read by automaton 2 in ℓ_2 (invariant of ℓ_2 and guards of the outgoing edges) have the same truth value for all the states (s_1, s_2) such that $s_1 \in S_1$ (resp. $s_1 \in S_1'$). In the sequel, we say that valuation V of s (resp. V' of s') satisfies constraint g, when the valuations of all states (s_1, s_2) in s (resp. in s') satisfy g. Assume now that for some reachable states (S_1, s_2) and (S_1', s_2'), $(S_1, s_2)\ \mathcal{R}\ (S_1', s_2')$.

Local Action. If $a \in \Sigma_2 \setminus \Sigma_1$ is enabled from (S_1, s_2), then, there is an associated edge in A_2, $\ell_2 \xrightarrow{g,a,r} p_2$ such that guard g is satisfied by V. Let g' be the guard on the corresponding outgoing edge $(\ell_{1,2}, \ell_2) \xrightarrow{g',a,r} (\ell_{1,2}, p_2)$ in A_2'. g uses clocks in X_2, and by construction, g' has the same form but with clocks in $(X_2 \setminus X_1) \uplus X_1'$. $(S_1, s_2)\ \mathcal{R}\ (S_1', s_2')$ says that v_2 and v_2' coincide on $X_2 \setminus X_1$, and since ☹ is never reached in $\mathcal{S}_{mod}$, V satisfies the constraints of g on X_1 iff

V' satisfies the constraints of g' on X'_1. That is, $V \models g \iff V' \models g'$. Therefore A'_2 can also perform a from (S_1, s'_2) and the states reached in both systems are $\mathcal{R}$-related: $(S_1, q_2)\ \mathcal{R}\ (S_1, q'_2)$, because $q_2 = (p_2, v_2[r])$ and $q'_2 = ((\ell_{1,2}, p_2), v'_2[r])$. This also holds reciprocally.

Synchronization. Assume for some $(a, s'_1) \in \mathbb{S} \times Q_1$, $(S_1, s_2) \xrightarrow{a,s'_1} (S'_1, q_2)$. That is, there is an edge $\ell_2 \xrightarrow{g_2,a,r_2} p_2$ in A_2 such that $v_2 \models g_2$ and $q_2 = (p_2, v_2[r_2])$ and, for some $(\ell_1, v_1) \in S_1$, an edge $\ell_1 \xrightarrow{g_1,a,r_1} p_1$ in A_1 such that $v_1 \models g_1$ and $s'_1 = (p_1, v_1[r_1]) \in S'_1$. Hence, synchronization $((a, p_1), s'_1)$ is also enabled from state (S_1, s'_2) because $A_{2,mod}$ is in the same location as A_2, and has the same clock values over $X_2 \setminus X_1$, and A'_1 is also in some state of S_1, therefore, there is also the same state $(\ell_1, v_1) \in S_1$ which enables (a, p_1). We do not consider $A_{1,2}$ because it is always ready to synchronize. Moreover, the state reached in $\psi(\mathrm{TTS}_{A'_1}(A'_2))$ after this synchronization is (S'_1, q'_2) such that $(S'_1, q_2)\ \mathcal{R}\ (S'_1, q'_2)$, because $q_2 = (p_2, v_2[r_2])$ and $q'_2 = \big((p_{1,2}, p_2), (v'_2[r_2])[c]\big)$ where c denotes the copy of the clocks of X_1 into their associated clocks of X'_1 and therefore c modifies only clocks that we do not consider in relation $\mathcal{R}$, and $r_2 \subseteq C_2 \subseteq (X_2 \setminus X_1)$ resets the same clocks in both systems. And reciprocally.

Local Delay. Assume for some $d \in \mathbb{R}_{\geq 0}$, $(S_1, s_2) \xrightarrow{d} (S'_1, q_2)$. Then, $V + d \models Inv_2(\ell_2)$, and since ☹ is never reached in $\mathcal{S}_{mod}$, $V + d \models Inv_2(\ell_2) \iff V' + d \models Inv'_2(\ell_2)$. That is, the same delay is enabled from (S_1, s'_2) while $A_{1,2}$ may perform some local steps: $(S_1, s'_2)(\xrightarrow{g_0,\varepsilon,r_0})^* \xrightarrow{d_0} (\xrightarrow{g_n,\varepsilon,r_n})^* \dots \xrightarrow{d_n} (S''_1, q'_2)$, where $\sum_{i=0}^{n} d_i = d$, g_i is a guards over X'_1 and r_i is a reset included in X'_1. This works because we assumed that A_1 has no urgent synchronization (and so does A'_1). Therefore, $A_{1,2}$ cannot force a synchronization.

Reciprocally, if we can perform a delay d from (S_1, s'_2), then $V' + d \models Inv'_2(\ell_2) \wedge Inv'_1(\ell_{1,2})$. And since $V + d \models Inv_2(\ell_2) \iff V' + d \models Inv'_2(\ell_2)$, we can perform the same delay from (S_1, s_2).

Moreover, we reach equivalent states in both systems. Indeed, A_2 and $A'_{2,mod}$ stay in the same location, the clocks in $X_2 \setminus X_1$ increase their value by d, and the set of states of A_1 and A'_1 becomes $S'_1 = S''_1 = \{s'_1 \mid \exists s_1 \in S_1, w \in \mathrm{TW}(\Sigma_1 \setminus \Sigma_2^{\not s}, d) : (s_1, s_2) \overset{w}{\Rightarrow} (s'_1, q_2)\}$.

Therefore, $\mathcal{R}$ is a weak timed bisimulation and $\psi(\mathrm{TTS}_{A'_1}(A'_2)) \sim \mathrm{TTS}_{A_1}(A_2)$. Lastly, by Lemma 2, $\psi(\mathrm{TTS}_{Q'_1}(A'_1 \parallel A'_2)) \sim \mathrm{TTS}_{Q_1}(A_1 \parallel A_2)$ also, and A_2 does not need to read X_1. ☐

In the example of Fig. 2, ☹ is not reachable in $\mathcal{S}_{mod}$ (see Fig. 6), therefore A_2 does not need to read X_1. For an example where ☹ is reachable, consider the same example with an additional edge $\xrightarrow{\top,f,\{x\}}$ from the end location of A_1 to a new location. Location ☹ can now be reached in $\mathcal{S}_{mod}$, for example consider a run where s is performed at time 2 leading to a state where $v(x) = 2$ and $v(x') = 2$, and then A_1 immediately performs f and resets x, leading to a state where the valuation v' is such that $v'(x) = 0$ and $v'(x') = 2$, and satisfies guard $x' \geq 1 \wedge x < 1$ in $\mathcal{S}_{mod}$. Therefore, with this additional edge in A_1, A_2 needs to read X_1. Indeed, without this edge, A_2 knows that A_1 cannot modify x after the synchronization, but with this edge, A_2 does not know whether A_1 has performed f and reset x, while this may change the truth value of its guard $x \geq 1$.

5.2 Complexity

PSPACE-hardness. The reachability problem for timed automata is known to be PSPACE-complete [AD90]. We will reduce this problem to our problem of deciding whether A_2 needs to read the clocks of A_1. Consider a timed automaton A over alphabet Σ, with some location ℓ.

Build the timed automaton A_2 as A augmented with two new locations ℓ' and ℓ'' and two edges, $\ell \xrightarrow{\top,\varepsilon,\emptyset} \ell'$ and $\ell' \xrightarrow{x=1,a,\emptyset} \ell''$, where x is a fresh clock, and a is some action in Σ. Let A_1 be the one of Fig. 4 with an action $b \notin \Sigma$. Then, ℓ is reachable in A iff A_2 needs to read x which belongs to A_1. Therefore the problem of deciding whether A_2 needs to read the clocks of A_1 is also PSPACE-hard.

PSPACE-membership. Moreover, we can show that when A_2 is deterministic, our problem is in PSPACE. Indeed, by Theorem 1 and Lemma 3, ☹ is not reachable iff $noRestriction_{A_1}(A_2)$ iff A_2 does not need to read the clocks of A_1. Since the size of the modified system on which we check the reachability of ☹ is polynomial in the size of the original system, our problem is in PSPACE.

5.3 Dealing with Urgent Synchronizations

If we use exactly the same construction as before and allow urgent synchronizations, the following problem may occur. Remind that $A_{1,2}$ simulates a possible run of A'_1 while A'_1 plays its actual run. There is no reason why the two runs should coincide. Thus it may happen that the run simulated by $A_{1,2}$ reaches a state where the invariant expires and only a synchronization is possible. Then A'_2 is expecting a synchronization with A'_1, but it is possible that the actual A'_1 has not reached a state that enables this synchronization. Intuitively, A'_2 should then realize that the simulated run cannot be the actual one and try another run compatible with the absence of synchronization.

But it is simpler to avoid this situation, which we can do by forcing $A_{1,2}$ to simulate one of the runs of A'_1 (from the state reached after the last synchronization) that has maximal duration before it synchronizes again with $A_{2,mod}$ (or never synchronizes again if possible). [1] This choice of a run of A'_1 is as valid as the others, and it prevents the system from having to deal with the subtle situation that we described above.

For example, consider automaton A_1 in Fig. 7 (it is the same as in Fig. 2 without the edge labeled by c and with guard $x \leq 1$ instead of $x < 1$). We can see that $A_{1,2}$ has to fire b at time 1 and is able to wait 3 time units before synchronizing, although it is still able to synchronize at any time (we add the same dashed edges as in Fig. 6). This can be generalized for any A_1. The idea is essentially to force $A_{1,2}$ to follow the appropriate finite or ultimately periodic path in the region automaton [AD94] of A_1. The construction is illustrated by Fig. 8 and 9.

More precisely, $A_{1,2}$ is now built over the region automaton [AD94] of A_1. Transitions labeled by some $a \in \mathbb{S}$ treated separately like in the original construction. The problem now is to constrain $A_{1,2}$ to take one of the most time consuming runs between two synchronizations.

Consider a state q in the region graph with synchronizations removed. If one of the paths from q has a loop, then there is an infinite execution from q with local actions, and since we considered non-Zeno TA, this implies that time diverges and we can impose this execution. If no path from q contains a loop, then these paths are finite and there is a finite number of such paths. It is possible to compute, for each path, the supremum of the time spent in this path (including the time spent in the last location) and select the largest one (which may be infinite). Assuming that this supremum is reached[2], we can impose one of the most time consuming runs

[1] There may not be any maximum if some time constraints are strict inequalities, but the idea can be adapted even to this case.

[2] If the supremum is finite and is not reached, then the construction can still be adapted. The idea is to follow the path until the last region with some possible timing. When it is reached, $A_{2,mod}$ can stop using the values of the clocks of $A_{1,2}$ to evaluate the truth value of its time constraints over clocks of A_1, but simply take their truth value according to the last region. These truth values can be used by $A_{2,mod}$ since they correspond to a path of

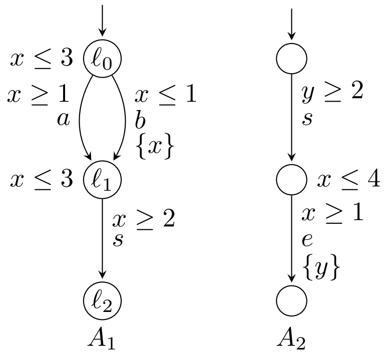

Figure 7: A_1 has an urgent synchronization

using a fresh clock and the appropriate guards and invariants.

Lastly, for each synchronizing edge in A_1, and each corresponding output state in the region automaton, we add synchronizing edges from all locations, to the location associated with this output state. These edges are labeled by "$\gamma, (a, \ell_1), c$", where γ is the constraint that describes the region associated with the target state, a is the synchronization label in A_1, ℓ_1 is the output location of the synchronization in A_1, and c is the copy of clock values.

Then we prove, in the same way as in Subsection 5.1, that, $A'_2 = A_{1,2} \otimes A'_{2,mod}$ is a suitable candidate even when urgent synchronization are allowed.

Indeed, with this construction, between two synchronizations, $A_{1,2}$ models one specific execution, σ'_1, of A_1. And if ☹ is never reached, then this means that any execution of A_1 is equivalent with this execution σ'_1, w.r.t. what A_2 tests. Hence, all executions of A_1 are equivalent w.r.t. what A_2 tests.

Finally, for automaton A_1 in Fig. 7, we get the region automaton of Fig. 8. After the synchronizations are removed, 6 final states can be reached from the initial state, with 6 possible paths. For each one of them, we compute the most time consuming one (we sum the maximal delays in each location, so that the path is possible and we add the maximal delay in the last location). All paths with action a have maximal duration of 3, and the path with action b has maximal duration of 4, when b is performed at time 1.

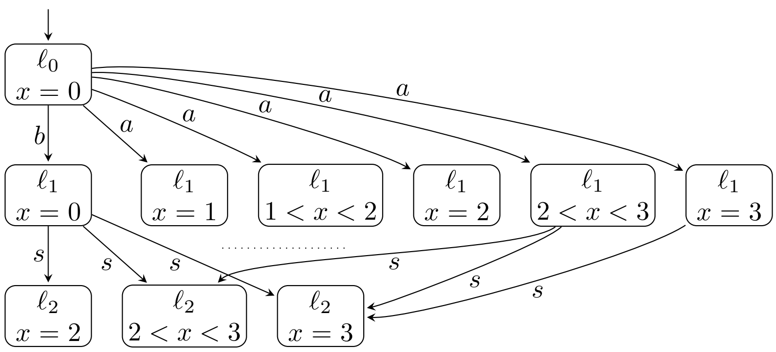

Figure 8: Region automaton of A_1 of Fig. 7

A_1 similar to but more time consuming than the simulated one.

Therefore, we impose the firing of b at time 1 in $A_{1,2}$, with adequate timing constraints, using a new clock, z. Lastly we get the automaton $A_{1,2}$ of Fig. 9.

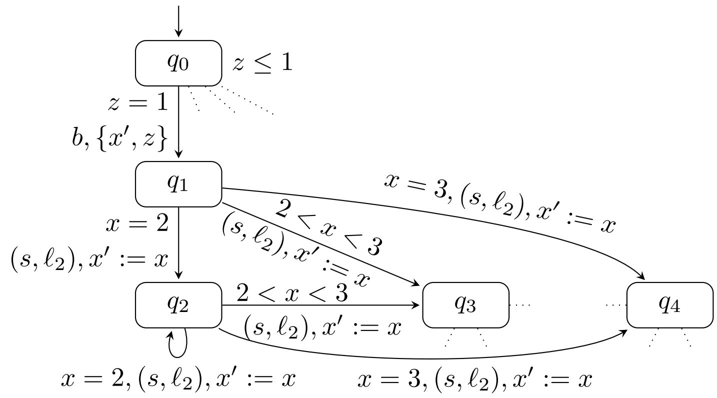

Figure 9: $A_{1,2}$ associated with A_1 of Fig. 7. Dotted lines denote edges that are not represented.

6 Conclusion

We have shown that in a distributed framework, when locality of actions and synchronizations matter, NTA with shared clocks cannot be easily transformed into NTA without shared clocks. The fact that the transformation is possible can be characterized using the notion of contextual TTS which represents the knowledge of one automaton about the other. Checking whether the transformation is possible is PSPACE-complete.

One conclusion is that, contrary to what happens when one considers the sequential semantics, NTA with shared clocks are strictly more expressive if we take distribution into account. This somehow justifies why shared clocks were introduced: they are actually more than syntactic sugar.

Another interesting point that we want to recall here, is the use of transmitting information during synchronizations. It is noticeable that infinitely precise information is required in general. This advocates the interest of updatable (N)TA used in an appropriate way, and more generally gives a flavor of a class of NTA closer to implementation.

Perspectives. Our first perspective is to generalize our result to the symmetrical case where A_1 also reads clocks from A_2. Then of course we can tackle general NTA with more than two automata.

Another line of research is to focus on transmission of information. The goal would be to minimize the information transmitted during synchronizations, and see for example where are the limits of finite information. Even when infinitely precise information is required to achieve the exact semantics of the NTA, it would be interesting to study how this semantics can be approximated using finitely precise information.

Finally, when shared clocks are necessary, one can discuss how to minimize them, or how to implement the model on a distributed architecture and how to handle shared clocks with as few communications as possible.

References

[ABG+08] S. Akshay, Benedikt Bollig, Paul Gastin, Madhavan Mukund, and K. Narayan Kumar. Distributed timed automata with independently evolving clocks. In Franck van Breugel and Marsha Chechik, editors, *CONCUR 2008*, volume 5201 of *LNCS*, pages 82–97. Springer, Heidelberg, 2008.

[AD90] Rajeev Alur and David Dill. Automata for modeling real-time systems. In Michael Paterson, editor, *ICALP 1990*, volume 443 of *LNCS*, pages 322–335. Springer, Heidelberg, 1990.

[AD94] Rajeev Alur and David Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.

[BC12] Sandie Balaguer and Thomas Chatain. Avoiding shared clocks in networks of timed automata. In Maciej Koutny and Irek Ulidowski, editors, *CONCUR 2012*, Lecture Notes in Computer Science, Newcastle, UK, September 2012. Springer, Heidelberg. To appear.

[BCH+05] Béatrice Bérard, Franck Cassez, Serge Haddad, Didier Lime, and Olivier H. Roux. Comparison of the expressiveness of timed automata and time Petri nets. In Paul Pettersson and Wang Yi, editors, *FORMATS 2005*, volume 3829 of *LNCS*, pages 211–225. Springer, Heidelberg, 2005.

[BCH12] Sandie Balaguer, Thomas Chatain, and Stefan Haar. A concurrency-preserving translation from time Petri nets to networks of timed automata. *FMSD*, 2012.

[BDFP04] Patricia Bouyer, Catherine Dufourd, Emmanuel Fleury, and Antoine Petit. Updatable timed automata. *Theor. Comput. Sci.*, 321(2-3):291–345, 2004.

[BDKP91] Eike Best, Raymond R. Devillers, Astrid Kiehn, and Lucia Pomello. Concurrent bisimulations in petri nets. *Acta Inf.*, 28(3):231–264, 1991.

[BDM+98] Marius Bozga, Conrado Daws, Oded Maler, Alfredo Olivero, Stavros Tripakis, and Sergio Yovine. Kronos: a model-checking tool for real-time systems. In Alan J. Hu and Moshe Y. Vardi, editors, *CAV 1998*, volume 1427 of *LNCS*, pages 546–550. Springer, Heidelberg, 1998.

[BDMP03] Patricia Bouyer, Deepak D'Souza, P. Madhusudan, and Antoine Petit. Timed control with partial observability. In Warren A. Hunt, Jr and Fabio Somenzi, editors, *CAV 2003*, volume 2725 of *LNCS*, pages 180–192. Springer, Heidelberg, 2003.

[BHR06] Patricia Bouyer, Serge Haddad, and Pierre-Alain Reynier. Timed unfoldings for networks of timed automata. In Susanne Graf and Wenhui Zhang, editors, *ATVA 2006*, volume 4218 of *LNCS*, pages 292–306. Springer, Heidelberg, 2006.

[BJLY98] Johan Bengtsson, Bengt Jonsson, Johan Lilius, and Wang Yi. Partial order reductions for timed systems. In *CONCUR 1998*, volume 1466 of *LNCS*, pages 485–500. Springer, Heidelberg, 1998.

[BR08] Marc Boyer and Olivier H. Roux. On the compared expressiveness of arc, place and transition time Petri nets. *Fundam. Inform.*, 88(3):225–249, 2008.

[CCJ06] Franck Cassez, Thomas Chatain, and Claude Jard. Symbolic unfoldings for networks of timed automata. In Susanne Graf and Wenhui Zhang, editors, *ATVA 2006*, volume 4218 of *LNCS*, pages 307–321. Springer, Heidelberg, 2006.

[CGL93] Karlis Cerans, Jens Chr. Godskesen, and Kim Guldstrand Larsen. Timed modal specification - theory and tools. In Costas Courcoubetis, editor, *CAV 1993*, volume 697 of *LNCS*, pages 253–267. Springer, Heidelberg, 1993.

[CR06] Franck Cassez and Olivier H. Roux. Structural translation from time Petri nets to timed automata. *Jour. of Systems and Software*, 2006.

[Dim09] Cătălin Dima. Positive and negative results on the decidability of the model-checking problem for an epistemic extension of timed ctl. In *TIME*, pages 29–36. IEEE Computer Society, 2009.

[DL07] Cătălin Dima and Ruggero Lanotte. Distributed time-asynchronous automata. In Cliff B. Jones, Zhiming Liu, and Jim Woodcock, editors, *ICTAC 2007*, LNCS, pages 185–200. Springer, Heidelberg, 2007.

[DLLN09] Alexandre David, Kim G. Larsen, Shuhao Li, and Brian Nielsen. Timed testing under partial observability. In *ICST*, pages 61–70. IEEE Computer Society, 2009.

[HFMV95] Joseph Y. Halpern, Ronald Fagin, Yoram Moses, and Moshe Y. Vardi. *Reasoning About Knowledge*. MIT Press, 1995.

[LNZ05] Denis Lugiez, Peter Niebert, and Sarah Zennou. A partial order semantics approach to the clock explosion problem of timed automata. *Theor. Comput. Sci.*, 345(1):27–59, 2005.

[LPY97] Kim G. Larsen, Paul Pettersson, and Wang Yi. Uppaal in a nutshell. *Jour. on Software Tools for Technology Transfer*, 1(1-2):134–152, 1997.

[MF76] P. M. Merlin and David J. Farber. Recoverability of communication protocols – implications of a theorical study. *IEEE Transactions on Communications*, 24, 1976.

[Min99] Marius Minea. Partial order reduction for model checking of timed automata. In Jos C. M. Baeten and Sjouke Mauw, editors, *CONCUR 1999*, volume 1664 of *LNCS, Heidelberg*, pages 431–446. Springer, 1999.

[Rei84] John Reif. The complexity of two-player games of incomplete information. *Jour. Computer and Systems Sciences*, 29:274–301, 1984.

[Srb08] Jirí Srba. Comparing the expressiveness of timed automata and timed extensions of Petri nets. In Franck Cassez and Claude Jard, editors, *FORMATS 2008*, volume 5215 of *LNCS*, pages 15–32. Springer, Heidelberg, 2008.

[vGG01] Rob J. van Glabbeek and Ursula Goltz. Refinement of actions and equivalence notions for concurrent systems. *Acta Inf.*, 37(4/5):229–327, 2001.

[WL04] Bozena Wozna and Alessio Lomuscio. A logic for knowledge, correctness, and real time. In João Alexandre Leite and Paolo Torroni, editors, *CLIMA 2004*, volume 3487 of *LNCS*, pages 1–15. Springer, Heidelberg, 2004.

RESEARCH CENTRE
SACLAY – ÎLE-DE-FRANCE

Parc Orsay Université
4 rue Jacques Monod
91893 Orsay Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399