Timed Modal Logics for Real-Time Systems - Specification, Verification and Control.

Patricia Bouyer, Franck Cassez, François Laroussinie

▶ **To cite this version:**

Patricia Bouyer, Franck Cassez, François Laroussinie. Timed Modal Logics for Real-Time Systems - Specification, Verification and Control.. Journal of Logic, Language and Computation, 2010. inria-00493638

HAL Id: inria-00493638
https://inria.hal.science/inria-00493638

Submitted on 21 Jun 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Timed Modal Logics for Real-Time Systems

Specification, Verification and Control

Patricia Bouyer*
Lab. Spécification & Vérification
ENS de Cachan & CNRS UMR 8643
Email: bouyer@lsv.ens-cachan.fr

Franck Cassez*†
National ICT Australia & CNRS
The University of New South Wales, Sydney, Australia
Email: franck.cassez@cnrs.irccyn.fr

François Laroussinie*
LIAFA, Institut Universitaire de France
Université Paris Diderot & CNRS UMR
Email: Francois.Laroussinie@liafa.jussieu.fr

Abstract. In this paper, a timed modal logic L_c is presented for the specification and verification of real-time systems. Several important results for L_c are discussed. First we address the model checking problem and we show that it is an EXPTIME-complete problem. Secondly we consider expressiveness and we explain how to express strong timed bisimilarity and how to build characteristic formulas for timed automata. We also propose a compositional algorithm for L_c model checking. Finally we consider several control problems for which L_c can be used to check controllability.

Keywords: Model checking, Timed automata, Timed modal logic, Timed control

1. Introduction

Model checking is widely used for the design and debugging of critical reactive systems [18, 17]. In the last fifteen years, it has been extended to *real-time systems*, where quantitative information about time is required. To express *quantitative* requirements of systems, we can use *timed logics* to write *timed specifications*.

Timed models. Real-time model checking has been mostly studied and developed in the framework of Alur and Dill's *Timed Automata* [5, 7], *i.e.*, automata extended with *dense-time clocks* that evolve synchronously with time. The behavior of a real-time system can be modeled as a parallel composition of timed automata. There now exists

* Partly supported by ANR project DOTS and project QUASIMODO (FP7-ICT).

† Author supported by a Marie Curie International Outgoing Fellowship within the 7th European Community Framework Programme.

© 2010 *Kluwer Academic Publishers. Printed in the Netherlands.*

a large body of theoretical knowledge and practical experience for this class of systems.

Timed specifications. Temporal and modal logics provide a fundamental framework for formally specifying systems and reasoning about them [18, 28]. For example, a property like

"Any problem is followed by an alarm." (1)

can be easily expressed with temporal/modal logics like *CTL*, *LTL*, μ-calculus. In the context of real-time systems, Property (1) can be better formulated as a *timed property* like:

"Any problem is followed by an alarm *within 10 time units*." (2)

In order to express such timing requirements, we can extend the classical temporal or modal logics. A first possibility is to use *timed* temporal modalities [9]. For example, with the timed version of *CTL*, namely TCTL, one can specify property (2) as follows:

$$\mathsf{AG}\Big(\mathsf{problem} \;\Rightarrow\; \mathsf{AF}_{\leq 10}\,\mathsf{alarm}\Big)$$

using a subscript for the temporal operators.

A more expressive method [14] consists in adding clocks — a.k.a. freeze variables — in the specification language [10, 14]. In this framework, a *formula clock* x can be reset (with the $\underline{\texttt{in}}$ operator) and compared against some constant later on. For example, the previous property can be equivalently written as follows with explicit formula clocks:

$$\mathsf{AG}\Big(\mathsf{problem} \;\Rightarrow\; x \;\underline{\texttt{in}}\; \mathsf{AF}(x \leq 10 \;\wedge\; \mathsf{alarm})\Big)$$

Such timing extensions carry over modal logics with fixpoint operators, and the point of this paper is to study such a logic called L_c.

Outline. The logic L_c is an extension of the timed modal logic L_ν introduced in [24, 21]. Those logics are natural extensions of classical modal logics for finite labeled transition systems w.r.t. the following aspects: 1) *fixpoint* operators allow to express a wide range of properties; 2) *timed* modalities ($[\delta\rangle_w$ and $[\delta\rangle_s$ in the sequel), allow existential/universal quantification over time delays; 3) *formula clocks* can be used in the formula to define quantitative time properties.

We address several issues related to the above-mentioned logics: complexity of model-checking, expressive power, and application to control problems. This paper is based on earlier results published in

the following papers [4, 21, 24, 23, 13]. Some results of section 5 are new, and they extend the previous results of [13] on L_ν^{det} to the logic L_c^{det} (see section 2.3).

2. Timed Automata and the Modal Logic L_c

We first recall the definition of *timed automata* proposed by Alur and Dill in [6] and then we introduce the timed modal logic L_c.

2.1. Notations

Let Act be a finite set of *actions*, and let $\mathbb{N}$, $\mathbb{Q}_+$ and $\mathbb{R}_+$ denote the sets of natural, non-negative rational and non-negative real numbers, respectively. Let X be a set of clocks. A *clock valuation* for X is a function from X to $\mathbb{R}_+$, the set of valuations is denoted $\mathbb{R}_+^X$. Given a valuation $v \in \mathbb{R}_+^X$ and $t \in \mathbb{R}_+$, $v+t$ is a valuation s.t. $(v+t)(x) = v(x)+t$ for any $x \in X$. Let $r \subseteq X$, $v[r \leftarrow 0]$ is the valuation defined by $v[r \leftarrow 0](x) = 0$ for any $x \in r$ and $v[r \leftarrow 0](x) = v(x)$ otherwise.

We denote by $\mathcal{C}(X)$ the set of *clock constraints* defined as the boolean combinations of atomic constraints of the form $x \sim p$ or $x - y \sim p$, with $x, y \in X$, $p \in \mathbb{N}$, and $\sim \in \{<, \leq, >, \geq, =\}$. Given $g \in \mathcal{C}(X)$ and $v \in \mathbb{R}_+^X$, we write $v \models g$ when v satisfies g. We write $\mathcal{C}_{\leqslant}(X)$ for the subset of $\mathcal{C}(X)$ which consist of the conjunctions of constraints of the form $x \leq p$ or $x < p$, with $p \in \mathbb{N}$.

2.2. Networks of Timed Automata

DEFINITION 1. *A* timed automaton *is a tuple* $\mathcal{A} = (L, \ell_0, \mathsf{Act}, X, \mathsf{Inv}, T)$ *where* L *is a finite set of* locations, $\ell_0 \in L$ *is the* initial *location,* X *is a finite set of* clocks, $\mathsf{Inv} : L \to \mathcal{C}_{\leqslant}(X)$ *is a function that assigns an* invariant *to each location, and* $T \subseteq L \times \mathcal{C}(X) \times \mathsf{Act} \times 2^X \times L$ *is a finite set of* edges. *A quintuple* $(\ell, g, a, r, \ell') \in T$ *represents an edge from location* ℓ *to location* ℓ' *with action* a, g *is the* guard *and* r *is a set of clocks to be reset to* 0. *In the sequel, we also use the notation* $\ell \xrightarrow{g,a,r} \ell'$ *for an egde.*

A guard is used to specify when a transition can be performed. An invariant is used to avoid excessive time delays in a location and thus it may enforce action transitions to be performed.

The semantics of a timed automaton s traditionally given by a timed transition system. A *timed transition system* (TTS for short) is a tuple $S = (Q, q_0, \mathsf{Act}, \rightarrow_S)$ where Q is a set of states, $q_0 \in Q$ is the initial

state, and $\rightarrow_S \subseteq Q \times (\mathsf{Act} \cup \mathbb{R}_+) \times Q$ is a set of transitions (again we write $q \xrightarrow{e}_S q'$ when $(q, e, q') \in \rightarrow_S$). The transitions labeled by $a \in \mathsf{Act}$ (resp. $t \in \mathbb{R}_+$) are called *action* (resp. *delay*) transitions. We make the following common assumptions about delay transitions in TTSs:

- 0-delay: $q \xrightarrow{0}_S q'$ if and only if $q = q'$,
- Time additivity: if $q \xrightarrow{t}_S q'$ and $q' \xrightarrow{t'}_S q''$ with $t, t' \in \mathbb{R}_+$, then $q \xrightarrow{t+t'}_S q''$,
- Time continuity: if $q \xrightarrow{t}_S q'$, then $\forall t', t'' \in \mathbb{R}_+$ with $t = t' + t''$, there exists q'' such that $q \xrightarrow{t'}_S q'' \xrightarrow{t''}_S q'$,
- Time determinism: if $q \xrightarrow{t}_S q'$ and $q \xrightarrow{t}_S q''$ with $t \in \mathbb{R}_+$, then $q' = q''$.

Standard notions of finite or infinite runs apply to TTS.

Given a timed automaton $\mathcal{A} = (L, \ell_0, \mathsf{Act}, X, \mathsf{Inv}, T)$, we define its semantics as the TTS $S_{\mathcal{A}} = (L \times \mathbb{R}_+^X, (\ell_0, v_0), \mathsf{Act}, \rightarrow_{S_{\mathcal{A}}})$ where $v_0(x) = 0$ for all $x \in X$ and $\rightarrow_{S_{\mathcal{A}}}$ consists of:

1. action transitions: $(\ell, v) \xrightarrow{a}_{S_{\mathcal{A}}} (\ell', v')$ if there exists an edge $\ell \xrightarrow{g,a,r} \ell'$ in T s.t. $v \models g$, $v' = v[r \leftarrow 0]$ and $v' \models \mathsf{Inv}(\ell')$;
2. delay transitions: $(\ell, v) \xrightarrow{t}_{S_{\mathcal{A}}} (\ell, v')$ if $t \in \mathbb{R}_+$, $v' = v + t$ and[1] $v' \models \mathsf{Inv}(\ell)$.

A *state* (or *configuration*) of a timed automaton $\mathcal{A}$ is a pair (ℓ, v), where ℓ is a location — or control state — and v is a clock valuation for X. Note that there are infinitely (and uncountably) many states in a timed automaton. However, a key point (for decidability) is the synchronous time elapsing: all clocks increase at the same speed.

The size of $\mathcal{A}$, denoted $|\mathcal{A}|$, is

$$|L| + |X| + \sum_{(\ell,g,a,r,\ell') \in T} |g| + \sum_{\ell} |\mathsf{Inv}(\ell)|$$

where the size of a constraint is its length (constants are encoded in binary).

REMARK 1. *We have restricted the clock constraints to use constants in the set* $\mathbb{N}$*. Constants in* $\mathbb{Q}_+$ *could be used, but in this case an "equivalent" timed automaton with constants in* $\mathbb{N}$ *can be obtained by scaling the rational constraints (and properties when a timed logic is involved), see [7, Lemma 4.1].*

[1] Due to the definition of invariants, this entails $v + t' \models \mathsf{Inv}(\ell)$ for any $0 \leq t' \leq t$.

We model real-time systems as parallel compositions of timed automata with n-ary synchronization functions. Let $\mathcal{A}_1, \ldots, \mathcal{A}_n$ be n timed automata with $\mathcal{A}_i = (L_i, \ell_{i,0}, \mathsf{Act}, X^i, \mathsf{Inv}_i, T_i)$ and $X^i \cap X^j = \varnothing$ for $i \neq j$. A *synchronization function* σ is a partial function $(\mathsf{Act} \cup \{\bullet\}) \times \ldots \times (\mathsf{Act} \cup \{\bullet\}) \to \mathsf{Act}$, where $\bullet$ denotes a distinguished no–action symbol. This is a synchronization function with renaming of synchronized actions. We write $(\mathcal{A}_1 | \ldots | \mathcal{A}_n)_\sigma$ for the parallel composition of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ w.r.t. synchronization function σ. A network configuration is a pair $(\overline{\ell}, v)$ where $\overline{\ell} = (\ell_1, \ldots, \ell_n)$ is a vector of locations and v is a valuation for $X = \bigcup_{1 \leq i \leq n} X^i$, *i.e.* the global set of clocks of the network.

The semantics of $(\mathcal{A}_1 | \ldots | \mathcal{A}_n)_\sigma$ can be defined as a TTS whose states are the configurations of the network and the transitions are given by the two following rules:

$$
\begin{array}{lll}
(\overline{\ell}, v) \xrightarrow{t} (\overline{\ell}, v+t) & \text{iff} & \forall i \in \{1, \ldots, n\}, (\ell_i, v_i) \xrightarrow{t} (\ell_i, v_i + t) \\
(\overline{\ell}, v) \xrightarrow{b} (\overline{\ell'}, v') & \text{iff} & \sigma(a_1, \ldots, a_n) = b \text{ and } \forall i \in \{1, \ldots, n\}, \\
 & & \left\{ \begin{array}{l} a_i \in \mathsf{Act} \text{ implies } (\ell_i, v_i) \xrightarrow{a_i} (\ell'_i, v'_i) \\ a_i = \bullet \text{ implies } (\ell'_i = \ell_i \ \wedge \ v'_i = v_i) \end{array} \right.
\end{array}
$$

where u_i denotes the restriction of u to the set of clocks X^i.

Note that the parallel composition does not add expressive power to timed automata: from any parallel composition of timed automata, one can build an *equivalent* (*i.e.* strongly bisimilar, see Section 4.1) timed automaton. Hence, in the following, unless specified, we will consider single timed automata.

2.3. The Timed Modal Logic L_c

2.3.1. *Syntax of L_c*

We define L_c, a timed modal μ-calculus that extends the logic L_ν [24] with "until"-style modalities.

DEFINITION 2. *Let K be a finite set of clocks (disjoint from X), and Id be a countably infinite set of identifiers (ranged over by X, Y). The set L_c of formulas over Act, K and Id is generated by the following grammar:*

$$
\begin{array}{rcl}
L_c \ni \varphi, \psi & ::= & \mathtt{tt} \mid \mathtt{ff} \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid r \ \underline{\mathtt{in}} \ \varphi \mid g \mid \\
 & & [a]\,\varphi \mid \langle a \rangle \varphi \mid \varphi\, [\delta\rangle_w \, \psi \mid \varphi\, [\delta\rangle_s \, \psi \mid Z
\end{array}
$$

where $a \in \mathsf{Act}$, $g \in \mathcal{C}(K)$, $r \subseteq K$ and $Z \in \mathsf{Id}$.

The meaning of the identifiers in Id is specified by a declaration $\mathcal{D}$ assigning an L_c formula to every identifier in order to define properties with maximal fixpoints. A declaration $\mathcal{D}$ is of the form $Z_i = \mathcal{D}(\overline{Z})$ where $\overline{Z} = (Z_1, \cdots, Z_n)$ and $\mathcal{D}(\overline{Z})$ is a formula over identifiers in $\overline{Z}$.

2.3.2. *Semantics of L_c*
Let $\mathcal{A} = (L, \ell_0, \mathsf{Act}, X, \mathsf{Inv}, T)$ be a timed automaton, and assume that the TTS $S_{\mathcal{A}} = (L \times \mathbb{R}_+^X, (\ell_0, v_0), \mathsf{Act}, \rightarrow_{S_{\mathcal{A}}})$ gives its semantics.

We interpret L_c formulas over extended states of the form (ℓ, v, u) where (ℓ, v) is a configuration of $\mathcal{A}$ and u is a valuation for the formula clocks in K. The formal semantics of L_c formulas interpreted over TTS $S_{\mathcal{A}}$ is given by the satisfaction relation $\models$ defined as the largest relation satisfying the implications in Table I (it can alternatively be defined with $\Leftrightarrow$ in place of $\Rightarrow$ and largest relation).

The intuition for the various operators are as follows. The modality $\langle a \rangle$ (resp. $[a]$) corresponds to existential (resp. universal) quantification over action transitions. The modalities $[\delta\rangle_w$ and $[\delta\rangle_s$ correspond to "weak until" and "strong until" over delay transitions: the time domain is dense, there is thus no notion of next state *via* these transitions. A configuration satisfies $\varphi\, [\delta\rangle_s\, \psi$ when there exists a future configuration satisfying ψ s.t. φ holds for all the intermediary configurations. For $[\delta\rangle_w$, it is not required to reach a configuration where ψ is true. An extended state satisfies an identifier Z if it belongs to the maximal fixpoint of the equation $Z = \mathcal{D}(Z)$. Finally the formula clocks are used to measure time elapsing in properties.

The timed automaton $\mathcal{A}$ satisfies φ if $(\ell_0, v_0, u_0) \models \varphi$ where $u_0(x) = 0$ for all $x \in K$.

We can easily define the standard modalities corresponding to existential and universal quantifications over delay transitions:

$$\langle\delta\rangle\ \varphi \ \stackrel{\text{def}}{=}\ \mathtt{tt}\ [\delta\rangle_s\ \varphi \tag{3a}$$

$$[\delta]\ \varphi \ \stackrel{\text{def}}{=}\ \varphi\ [\delta\rangle_w\ \mathtt{ff} \tag{3b}$$

Moreover we also have the following equivalences:

$$\varphi\, [\delta\rangle_s\, \psi \ \equiv\ \varphi\, [\delta\rangle_w\, \psi \wedge \langle\delta\rangle\ \psi \tag{4a}$$

$$\varphi\, [\delta\rangle_w\, \psi \ \equiv\ \varphi\, [\delta\rangle_s\, \psi \vee [\delta]\ \varphi \tag{4b}$$

2.3.3. *Other timed modal logics*
The timed modal logic L_ν [24] is the restriction of L_c where the modalities $[\delta\rangle_w$ and $[\delta\rangle_s$ are not used, and instead, only $[\delta]$ and $\langle\delta\rangle$ can be used. Several restrictions of L_ν, namely L_s ("Logic for Safety"), *SBLL*

Table I. Satisfaction implications for L_c

$(\ell, v, u) \models \mathtt{tt} \Rightarrow \mathtt{tt}$
$(\ell, v, u) \models \mathtt{ff} \Rightarrow \mathtt{ff}$
$(\ell, v, u) \models g \Rightarrow u \models g$
$(\ell, v, u) \models \varphi_1 \vee \varphi_2, \Rightarrow (\ell, v, u) \models \varphi_1$ or $(\ell, v, u) \models \varphi_2$
$(\ell, v, u) \models \varphi_1 \wedge \varphi_2, \Rightarrow (\ell, v, u) \models \varphi_1$ and $(\ell, v, u) \models \varphi_2$
$(\ell, v, u) \models r \ \underline{\mathtt{in}} \ \varphi \Rightarrow (\ell, v, u[r \leftarrow 0]) \models \varphi$
$(\ell, v, u) \models [a] \, \varphi \Rightarrow$ for all $(\ell, v) \xrightarrow{a}_{S_{\mathcal{A}}} (\ell', v')$, $(\ell', v', u) \models \varphi$
$(\ell, v, u) \models \langle a \rangle \, \varphi \Rightarrow$ there is some $(\ell, v) \xrightarrow{a}_{S_{\mathcal{A}}} (\ell', v')$, s.t. $(\ell', v', u) \models \varphi$
$(\ell, v, u) \models \varphi \, [\delta\rangle_s \, \psi \Rightarrow$ there is some $t \in \mathbb{R}_+$ s.t. $(\ell, v) \xrightarrow{t}_{S_{\mathcal{A}}} (\ell, v + t)$,
$(\ell, v + t, u + t) \models \psi$, and for every $0 \leq t' < t$, $(\ell, v + t', u + t') \models \varphi$
$(\ell, v, u) \models \varphi \, [\delta\rangle_w \, \psi \Rightarrow$ for all $t \in \mathbb{R}_+$ s.t. $(\ell, v) \xrightarrow{t}_{S_{\mathcal{A}}} (\ell, v + t)$,
$(\ell, v + t, u + t) \models \varphi$ or $(\ell, v, u) \models \varphi \, [\delta\rangle_s \, \psi$
$(\ell, v, u) \models Z \Rightarrow (\ell, v, u) \models \mathcal{D}(Z)$

(Safety and Bounded Liveness Logic") and $L_{\forall S}$, have been considered in [25, 2, 1, 4].

In [19] a timed μ-calculus (T_μ) with minimal and maximal fixpoints has been defined. It contains a modality $\rhd$ whose semantics is close to the one of $[\delta\rangle_s$ (the main difference between $\rhd$ and $[\delta\rangle_s$ is that $\rhd$ may include an action transition after the delay).

2.3.4. *Examples of L_c properties*
We can use L_c formulas to express classical temporal (and timed) properties:

– The formula

$$z \ \underline{\mathtt{in}} \ \langle \delta \rangle \, (z < 10 \wedge (\langle a \rangle \, \mathtt{tt} \vee \langle b \rangle \, \mathtt{tt}))$$

specifies that an a-action or a b-action is enabled before 10 time units has elapsed.

– We can express that some formula φ holds at any reachable state ("ALWAYS_φ", corresponding to the $\mathsf{AG}\varphi$ formula of CTL). This can be defined by the following equation:

$$Z \stackrel{\text{def}}{=} \varphi \wedge \bigwedge_{a \in \mathsf{Act}} [a] \, Z \wedge [\delta] \, Z.$$

– Given two subsets of events Act_1 and Act_2, we can state that any event in Act_1 is followed by an event in Act_2 within less than Δ

time units (fixing some $\Delta \in \mathbb{N}$):

$$Z_1 \stackrel{\text{def}}{=} \left(\bigwedge_{a \in \mathsf{Act}_1} [a]\,(y \mathrel{\underline{\mathtt{in}}} Z_2) \right) \wedge \left(\bigwedge_{a \in \mathsf{Act} \setminus \mathsf{Act}_1} [a]\, Z_1 \right) \wedge [\delta]\, Z_1$$

$$Z_2 \stackrel{\text{def}}{=} (y < \Delta) \wedge \left(\bigwedge_{a \in \mathsf{Act}_2} [a]\, Z_1 \right) \wedge \left(\bigwedge_{a \in \mathsf{Act} \setminus \mathsf{Act}_2} [a]\, Z_2 \right) \wedge [\delta]\, Z_2.$$

– To express that some property φ will hold during at least Δ time units, whatever the transitions performed ($\varphi\ \mathsf{UpTo}\ \Delta$), we can use the L_c formula $z \mathrel{\underline{\mathtt{in}}} Z_1$ with:

$$Z_1 \stackrel{\text{def}}{=} (z > \Delta) \vee \left(\varphi \wedge \bigwedge_{a \in \mathsf{Act}} [a]\, Z_1 \wedge [\delta]\, Z_1 \right).$$

– We can express the *weak* until property $\mathsf{E}\varphi\mathsf{W}\psi$: there exists a path such that either φ holds until ψ, or the path is infinite (*i.e.,* with an infinite number of action transitions) and satisfies φ everywhere. This property can be defined by the following equation:

$$Z \stackrel{\text{def}}{=} \varphi\, [\delta\rangle_s \left(\psi \vee \bigvee_{a \in \mathsf{Act}} \langle a \rangle\, Z \right).$$

3. Model Checking and Satisfiability for the Logic L_c

3.1. Reachability and Model Checking for TCTL

Automatic verification of timed systems modelled as (networks of) timed automata is possible despite the uncountable number of states associated with the semantics of a timed automaton. Given a timed automaton $\mathcal{A}$ and a property φ, the decision procedure for the problem $\mathcal{A} \models \varphi$ is based on the well-known *region abstraction* (see [5] for a description of the algorithm for TCTL, a standard branching-time logic for real-time systems).

Indeed, it is possible to partition the uncountable set of valuations over $X \cup K$ into a finite number of *regions*, s.t. two extended configurations (ℓ, v, u) and (ℓ, v', u'), where uv and $u'v'$ are in the same region, satisfy the same subformulas of φ. Let c_{max} be the maximal constant occurring in the guards and invariants of the timed automaton $\mathcal{A}$ and in the formula φ. The regions can be defined as the equivalence classes

induced by the equivalence relation over valuations defined as follows: two valuations $w, w' \in \mathbb{R}_+^{X \cup K}$ are in the same region iff they satisfy the same clock constraints from the set $\mathcal{C}_{c_{max}}(X \cup K)$ containing $\mathcal{C}(X \cup K)$ expressions whose integral constants belong to $\{0, \ldots, c_{max}\}$. Due to this bound on the constants, the number of equivalence classes is clearly finite.

Using the regions, we can define a symbolic semantics for $\mathcal{A}$ which is a finite transition system, called the *region graph*, whose states are pairs (ℓ, γ) where ℓ is a location of $\mathcal{A}$ and γ a region. Formulas in L_c can then be interpreted over the states of the region graph (as it is done for L_ν in [24]).

The region graph technique provides decidability results for many verification problems over timed systems by reducing them to similar problems on finite graphs. Note however that the number of regions is exponential in the number of clocks and the maximal constant, it is more precisely in $O(|X \cup K|! \cdot c_{max}^{|X \cup K|})$. Consequently, the size of the region graph is in $O((|L| + |T|) \cdot |X \cup K|! \cdot c_{max}^{|X \cup K|})$.

THEOREM 1 ([7]). *The reachability and TCTL model checking problems are PSPACE-complete for timed automata.*

REMARK 2. *In practice, the region graph is not built, and tools like* Uppaal *[26] or* Kronos *[32], build a coarser graph, yet sufficient for verifying basic properties. The practical algorithms also use efficient data-structure based on the DBMs (Difference Bound Matrices) that allow to handle convex sets of valuations. Moreover several heuristics have been developed to improve the efficiency of the algorithms (on-the-fly algorithms, active clock reduction, etc.).*

3.2. Model Checking L_c

The results in [4] obtained for model checking L_ν extend to L_c and thus:

THEOREM 2. *The model checking problem for L_c is EXPTIME-complete. The specification and program complexities are also EXPTIME-complete.*

PROOF 1 (Sketch). *The EXPTIME membership comes from the fact that applying standard verification algorithms for modal logics over the region graph can be done in time linear in the size of the formula and in the size of the region graph (considering $[\delta\rangle_w$ and $[\delta\rangle_s$ instead*

of classical $\langle\delta\rangle$ and $[\delta]$ does not change anything). This provides an algorithm which is exponential in the size of the timed automaton and in the size of the formula[2]*.*

The EXPTIME-hardness comes from EXPTIME-hardness of L_ν (proved in [4]) which is a sublogic of L_c. We just sketch the proof (See [4] for the details) which consists in reducing the acceptance of a word w by a linear (space) bounded alternating Turing machine $\mathcal{M}$ to a model checking problem for L_ν. We assume (w.l.o.g.) that there is a strict alternation of existential and universal states in $\mathcal{M}$ and that the initial state is existential. First one can build a timed automaton $\mathcal{A}_{\mathcal{M},w}$ that represents the behavior of the non-alternating version of $\mathcal{M}$ over the word w (clocks are used to encode the contents of the $|w|$ cells of the tape): any action transition corresponds to a step of $\mathcal{M}$ and between two actions one requires a strictly positive delay. We distinguish three labels for actions of $\mathcal{M}$: Init corresponds to the writing of w on the tape at the beginning of the computation, a labels any step of $\mathcal{M}$ and $\mathsf{Accepting}$ labels accepting states of $\mathcal{M}$. Secondly one can use the L_ν formula $\Phi_{na} \stackrel{\text{def}}{=} [\delta]\,[\mathsf{Init}]\,Y$ with:

$$Y \stackrel{\text{def}}{=} [\mathsf{Accepting}]\,\mathtt{ff} \wedge [\delta]\,[\mathsf{a}]\,(\langle\delta\rangle\,\langle\mathsf{a}\rangle\,Y).$$

It can be proved that the formula Φ_{na} holds for the initial state of $\mathcal{A}_{\mathcal{M},w}$ iff $\mathcal{M}$ does not accept w.

The main difference with reachability or TCTL model checking for TA (PSPACE-complete problems) is the ability for L_c to simulate the alternating behavior of $\mathcal{M}$.

From the previous construction, one can easily deduce that the program complexity is also EXPTIME-hard since the formula used in the reduction does not depend on $\mathcal{M}$ and w.

Finally the specification complexity comes from the ability of L_ν (and L_c) to encode the behavior of a timed automaton. The previous reduction could have been done for a simple automaton with no clock and no edge and with a more complex formula. This can be obtained as a direct consequence of the properties of L_c presented in the next section. □

3.3. Satisfiability of L_c

The satisfiability problem for L_c asks the following: given an L_c formula Φ, does there exist a timed automaton $\mathcal{A}$ s.t. $\mathcal{A} \models \Phi$?

[2] Notice that the region graph is exponential in the number of automata and formula clocks.

The status of the satisfiability problem is still unknown for both L_c and L_ν [24]. Note that for many of the timed branching-time temporal logics (like TCTL, T_μ) the satisfiability problem is undecidable [5, 8, 19]. For the timed linear-time temporal logics like MTL, satisfiability is undecidable for the (standard) interval-based semantics [19], but is decidable (though non primitive recursive) for the pointwise semantics over finite timed words [30].

Finally, satisfiability under *bounded resources* (automata with a fixed number of clocks and values of the constants) is decidable for L_ν [24] and this algorithm can easily be extended to L_c.

4. Expressiveness of L_c

We consider several expressiveness properties of L_ν and L_c: strong timed bisimilarity and characteristic formulas are investigated in Sections 4.1 and 4.3; Compositionality is presented in Section 4.2 and is an interesting property of L_c, which enables one to check an L_c property by computing a *quotient* formula; Finally, in Section 4.4, we compare the relative expressive power of L_c and L_ν.

4.1. Strong Timed Bisimilarity

The standard notion of bisimulation [29, 31] can be naturally extended to timed systems. In the context of timed systems, several bisimulation relations are of interest, like the time-abstract bisimulation or the strong timed bisimulation, and in this paper we focus on the latter.

Let $S_A = (Q_A, q_0^A, \mathsf{Act}, \rightarrow_{S_A})$ and $S_B = (Q_B, q_0^B, \mathsf{Act}, \rightarrow_{S_B})$ be two TTSs over the same set of actions Act. Let $\sim$ be a symmetric relation over $Q_A \times Q_B \cup Q_B \times Q_A$. We say that this is a *strong timed bisimulation* whenever the following transfer property holds: if $q_A \sim q_B$, then

1. for every $q_A \xrightarrow{a}_{S_A} q'_A$ with $a \in \mathsf{Act}$, there exists $q_B \xrightarrow{a}_{S_B} q'_B$ such that $q'_A \sim q'_B$,
2. for every $q_A \xrightarrow{t}_{S_A} q'_A$ with $t \in \mathbb{R}_+$, there exists $q_B \xrightarrow{t}_{S_B} q'_B$ such that $q'_A \sim q'_B$.

The two TTSs S_A and S_B are said *strongly timed bisimilar* whenever there exists a strong timed bisimulation relation $\sim$ such that furthermore $q_0^A \sim q_0^B$. Two timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are *strongly timed bisimilar* if their corresponding TTSs are strongly timed bisimilar. In that case, we write $\mathcal{A}_1 \cong \mathcal{A}_2$.

In the following, we show that L_ν can express strong timed bisimilarity of timed automata.

Let $\mathcal{A}_i = (L_i, \ell_{i,0}, \mathsf{Act}, X^i, \mathsf{Inv}_i, T_i)$ for $i = 1, 2$ be two timed automata. Define $\mathsf{Act}_{1,2} = \{a_i \mid a \in \mathsf{Act}, i = 1, 2\}$. Let σ be the binary synchronization function defined as follows: for every $a \in \mathsf{Act}$, we have $\sigma(a, \bullet) = a_1$ and $\sigma(\bullet, a) = a_2$. With σ, the action transitions of the timed automata are not synchronized in $(\mathcal{A}_1|\mathcal{A}_2)_\sigma$ and they are renamed with a letter from $\mathsf{Act}_{1,2}$ containing the number of the timed automaton that is making the product transition. Note however that the delay transitions are still synchronized: all clocks of the composition evolve synchronously with time.

Now assume $\mathcal{A}_1$ and $\mathcal{A}_2$ have no invariant *i.e.,* for every location ℓ, $\mathsf{Inv}_i(\ell) = \mathtt{tt}$ (true). Let Z be defined by the following declaration (fixpoint equation):

$$Z \stackrel{\text{def}}{=} \bigwedge_{a \in \mathsf{Act}} \Big([a_1] \langle a_2 \rangle Z \wedge [a_2] \langle a_1 \rangle Z \Big) \wedge [\delta] Z. \qquad (\mathit{Bisim})$$

Checking whether $\mathcal{A}_1$ and $\mathcal{A}_2$ are strongly timed bisimilar can be done using formula (*Bisim*):

THEOREM 3 ([22]). *$\mathcal{A}_1 \cong \mathcal{A}_2$ iff $(\mathcal{A}_1|\mathcal{A}_2)_\sigma \models Z$.*

Note that the definition of Z is precisely the definition of the strong timed bisimilarity. Moreover all these results can be easily extended to deal with parallel compositions of timed automata.

When the automata have invariants, the parallel composition $(\mathcal{A}_1|\mathcal{A}_2)_\sigma$ may restrict some behaviors of $\mathcal{A}_1$ and/or $\mathcal{A}_2$: from a configuration $((\ell_1, \ell_2), v_1 v_2)$ of $(\mathcal{A}_1|\mathcal{A}_2)_\sigma$, a delay transition $\xrightarrow{t}$ is enabled iff it is enabled by both automata. If ℓ_1 has an invariant, the parallel composition does not contain delays that violate this invariant even if such delays exist in $\mathcal{A}_2$: hence the product $(\mathcal{A}_1|\mathcal{A}_2)_\sigma$ and Z cannot be used as before. To overcome this problem, we define two new timed automata $\mathcal{A}'_i = ((L_i, \ell_{i,0}, \mathsf{Act} \cup \{\mathsf{InvFail}\}, X^i, \mathsf{Inv}'_i, T'_i))$ where $\mathsf{InvFail}$ is a fresh action, Inv'_i assigns true to every location, and the set of edges T'_i is defined as follows:

- *violation of invariant:* for every location $\ell \in L_i$, there is an edge $(\ell, \neg \mathsf{Inv}_i(\ell), \mathsf{InvFail}, \varnothing, \ell)$ in T'_i;
- *strengthened edges:* every edge (ℓ, g, a, r, ℓ') in T_i is replaced by a stronger edge $(\ell, g \wedge \mathsf{Inv}_i(\ell) \wedge (\mathsf{Inv}_i(\ell')[r \leftarrow 0]), r, \ell')$ in T'_i, where $g[r \leftarrow 0]$ denotes the constraint g in which every occurrence of the clocks in r are replaced by the constant 0.

The first type of edges, labelled with the distinguished $\mathsf{InvFail}$ action, allows a timed automaton to fire an edge even if the original invariant

is false. The second type of edges corresponds to regular action steps: we add the requirement that in the guard, the current invariant has to be satisfied, and also that the invariant of the target location has to be satisfied by the valuation after the reset (we restrict to transitions that can effectively be taken). Using this construction, we have the following result:

THEOREM 4. *$\mathcal{A}_1 \cong \mathcal{A}_2$ iff $(\mathcal{A}'_1|\mathcal{A}'_2)_\sigma \models Z'$ with*

$$Z' \stackrel{\text{def}}{=} \bigwedge_{a \in \mathsf{Act} \cup \{\mathsf{InvFail}\}} \Big([a_1] \langle a_2 \rangle Z' \wedge [a_2] \langle a_1 \rangle Z' \Big) \wedge [\delta] Z'. \qquad (\textit{Bisim'})$$

PROOF 2. *A configuration (ℓ, v) of $\mathcal{A}_i$ is said* correct *whenever $v \models \mathsf{Inv}_i(\ell)$. Let $\sim$ be the largest strong timed bisimulation relation between $\mathcal{A}_1$ and $\mathcal{A}_2$. Consider two correct configurations (ℓ_1, v_1) and (ℓ_2, v_2) of $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively. We prove that the following equivalence holds:*

$$(\ell_1, v_1) \sim (\ell_2, v_2) \quad \Leftrightarrow \quad ((\ell_1, \ell_2), v_1 v_2)_{(\mathcal{A}'_1|\mathcal{A}'_2)_\sigma} \models Z'.$$

Proof of $\Rightarrow$. *For $k \in \mathbb{N}$, we write $(\ell_1, v_1) \sim_k (\ell_2, v_2)$ to state that these configurations are strongly timed bisimilar up to depth k (*i.e., *for at least k action or delay transitions). We have $(\ell_1, v_1) \sim_0 (\ell_2, v_2)$ for correct configurations.*

Now let Z^0 be tt *and let Z^{k+1} be the formula:*

$$\bigwedge_{a \in \mathsf{Act} \cup \{\mathsf{InvFail}\}} \Big([a_1] \langle a_2 \rangle Z^k \wedge [a_2] \langle a_1 \rangle Z^k \Big) \wedge [\delta] Z^k.$$

We can prove by induction over k that $(\ell_1, v_1) \sim_k (\ell_2, v_2)$ implies $((\ell_1, \ell_2), v_1 v_2)_{(\mathcal{A}'_1|\mathcal{A}'_2)_\sigma} \models Z^k$. This is trivial for the basic case ($k = 0$), we now focus on the induction step (from k to $k+1$).

In the following we remove the subscript "$(\mathcal{A}'_1|\mathcal{A}'_2)_\sigma$" to configurations, and we thus implicitly assume that configurations refer to those of the parallel composition $(\mathcal{A}'_1|\mathcal{A}'_2)_\sigma$.

Assume $(\ell_1, v_1) \sim_{k+1} (\ell_2, v_2)$. Consider the subformula $[a_1] \langle a_2 \rangle Z^k$ and assume $((\ell_1, \ell_2), v_1 v_2) \xrightarrow{a_1} ((\ell'_1, \ell_2), v'_1 v_2)$. As (ℓ_1, v_1) is a correct configuration, we have that $v_1 \models \mathsf{Inv}_1(\ell_1)$ and $(\ell_1, v_1) \xrightarrow{a} (\ell'_1, v'_1)$ in $\mathcal{A}_1$ with $v'_1 \models \mathsf{Inv}_1(\ell'_1)$. As $(\ell_1, v_1) \sim_{k+1} (\ell_2, v_2)$, there exists $(\ell_2, v_2) \xrightarrow{a} (\ell'_2, v'_2)$ in $\mathcal{A}_2$ such that $(\ell'_1, v'_1) \sim_k (\ell'_2, v'_2)$. Invariants in $\mathcal{A}_2$ are properly satisfied ($v_2 \models \mathsf{Inv}_2(\ell_2)$ because (ℓ_2, v_2) is a correct configuration, and $v'_2 \models \mathsf{Inv}_2(\ell'_2)$ because it results from a move in $\mathcal{A}_2$), and there exists thus a move $((\ell'_1, \ell_2), v'_1 v_2) \xrightarrow{a_2} (\ell'_1, \ell'_2), v'_1 v'_2)$ in $(\mathcal{A}'_1|\mathcal{A}'_2)_\sigma$. By induction hypothesis we deduce that $((\ell'_1, \ell'_2), v'_1 v'_2) \models Z^k$. The same reasoning can be applied to the subformulas $[a_2] \langle a_1 \rangle Z^k$.

Now consider the subformula $[\delta]\ Z^k$ *and a delay transition* $((\ell_1,\ell_2),v_1v_2) \xrightarrow{t} ((\ell_1,\ell_2),(v_1v_2)+t)$. *We have to distinguish between several cases:*

- *there is no* $\mathsf{InvFail}_i$*-transition enabled from the new configuration.*[3] *Thus the two invariants are satisfied while delaying, and there exist delay steps* $(\ell_1,v_1) \xrightarrow{t} (\ell_1,v_1+t)$ *and* $(\ell_2,v_2) \xrightarrow{t} (\ell_2,v_2+t)$ *in the original automata* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *respectively. Then,* $(\ell_1,v_1+t) \sim_k (\ell_2,v_2+t)$ *ensures (by induction hypothesis) that* $((\ell_1,\ell_2),(v_1v_2)+t) \models Z^k$.
- *there is an* $\mathsf{InvFail}_1$*-transition and an* $\mathsf{InvFail}_2$*-transition enabled from the new configuration. By construction of the* $\mathcal{A}'_i$*'s, only actions* $\mathsf{InvFail}_i$ *will be enabled afterwards, and they will be enabled with no constraint. Thus clearly,* Z^k *holds.*
- *the last case,* $\mathsf{InvFail}_1$ *is enabled and* $\mathsf{InvFail}_2$ *is not enabled (or the converse) cannot occur because this would mean that* $\mathcal{A}_2$ *can perform a delay transition which is not possible from* $\mathcal{A}_1$ *and this is a contradiction with* $(\ell_1,v_1) \sim_{k+1} (\ell_2,v_2)$.

This clearly entails $((\ell_1,\ell_2),v_1v_2) \models Z^{k+1}$.

We now notice that $(\ell_1,v_1) \sim (\ell_2,v_2)$ *iff for every* $k \in \mathbb{N}$, $(\ell_1,v_1) \sim_k (\ell_2,v_2)$. *Applying the previous equivalence, we have that* $(\ell_1,v_1) \sim (\ell_2,v_2)$ *iff* $((\ell_1,\ell_2),v_1v_2) \models Z'$.

Proof of $\Leftarrow$. *We show that the following relation* $\mathcal{R}$ *over the correct configurations of* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *is a strong timed bisimulation. The relation* $\mathcal{R}$ *is defined as follows:* $(\ell_1,v_1)\mathcal{R}(\ell_2,v_2)$ *iff* $((\ell_1,\ell_2),v_1v_2)_{(\mathcal{A}'_1|\mathcal{A}'_2)_\sigma} \models Z'$. *We show that it satisfies the transfer property.*

Assume $(\ell_1,v_1)\mathcal{R}(\ell_2,v_2)$ *and consider a transition* $(\ell_1,v_1) \xrightarrow{a} (\ell'_1,v'_1)$ *in* $\mathcal{A}_1$. *As* $((\ell_1,\ell_2),v_1v_2) \models [a_1]\langle a_2\rangle Z'$, *there exists a transition* $((\ell'_1,\ell_2),v'_1v_2) \xrightarrow{a_2} ((\ell'_1,\ell'_2),v'_1v'_2)$ *and this last configuration satisfies* Z'. *This ensures* $(\ell'_1,v'_1)\mathcal{R}(\ell'_2,v'_2)$. *The same holds for a transition from* (ℓ_2,v_2).

Consider a transition $(\ell_1,v_1) \xrightarrow{t} (\ell_1,v_1+t)$ *in* $\mathcal{A}_1$. *We know that we have* $((\ell_1,\ell_2),(v_1v_2)+t) \models Z'$. *Thus the configuration satisfies* $[\mathsf{InvFail}_2]\langle\mathsf{InvFail}_1\rangle Z$: *as the invariant* $\mathit{Inv}_1(\ell_1)$ *is satisfied by* v_1+t, *we can deduce that there is no enabled transition labeled with* $\mathsf{InvFail}_2$ *and then* $v_2 \models \mathit{Inv}_2(\ell_2)$. *Thus the delay transition* $(\ell_2,v_2) \xrightarrow{t} (\ell_2,v_2+t)$ *is enabled in* $\mathcal{A}_2$. *Moreover the new configuration of the parallel composition satisfies* Z', *and then* $(\ell_1,v_1+t)\mathcal{R}(\ell_2,v_2+t)$. *Hence* $\mathcal{R}$ *is a strongly timed bisimulation relation.* □

[3] Note that in that case no action $\mathsf{InvFail}_i$ has been enabled while delaying. This is due to the form of the invariants, which are supposed to be constraints in $\mathcal{C}_{\leqslant}(X)$.

4.2. Compositionality

Compositional model checking [11] is suitable for verifying L_c properties: given a parallel composition $(\mathcal{A}_1|\ldots|\mathcal{A}_n)_\sigma$ and some L_c formula Φ, one can build a *quotient* formula $\Phi/\mathcal{A}_1$ such that $(\mathcal{A}_1|\ldots|\mathcal{A}_n)_\sigma \models \Phi$ iff $(\mathcal{A}_2|\ldots|\mathcal{A}_n)_\sigma \models \Phi/\mathcal{A}_1$. The formula $\Phi/\mathcal{A}_1$ integrates the initial property and the pertinent part w.r.t. Φ of the behavior of $\mathcal{A}_1$. By repeatedly quotienting components from the network into the formula, we will finally be left with the verification problem: check whether $nil \models \Phi/\mathcal{A}_1/\ldots/\mathcal{A}_n$ where *nil* is a timed automaton unable to perform any action transition (it can just let time elapse).

Table II presents the definition[4] of the quotient formulas for L_c [21, 23, 13] for a network of two timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$. Note that the quotient is defined for a formula φ, a location ℓ of a timed automaton, and a synchronization function σ. In the definition, we assume $\langle\bullet\rangle\,\varphi = [\bullet]\,\varphi = \varphi$. Note also that the clocks of the quotiented automaton becomes formula clocks in the quotient formula. We use $\varphi/\mathcal{A}$ to denote the quotiented formula φ/ℓ_0 where ℓ_0 is the initial location of $\mathcal{A}$. Finally any fixpoint variable of Φ gives rise to $|L|$ variables in $\Phi/\mathcal{A}$. The following theorem extends the result for L_ν [23] to L_c:

THEOREM 5. *Given two timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$, a synchronization function σ and an L_c formula φ, we have the following property for any configuration $((\ell_1,\ell_2),v_1v_2)$ of $(\mathcal{A}_1|\mathcal{A}_2)_f$ and $u \in \mathbb{R}_+^K$:*

$$((\ell_1,\ell_2),v_1v_2,u) \models \varphi \quad \textit{iff} \quad (\ell_2,v_2,uv_1) \models \varphi/\ell_1.$$

PROOF 3. *To prove this result we first show that the property holds for any formula without any fixpoint variable. In this case the proof is done by structural induction over the formula. The basic cases are straightforward and we just consider the modalities $\langle a\rangle$ and $[\delta\rangle_w$:*

- *Assume $\Psi \stackrel{\text{def}}{=} \langle a\rangle\,\varphi$ and $((\ell_1,\ell_2),v_1v_2,u) \models \Psi$. Then there exist $b,c \in \mathsf{Act}$ such that $\sigma(b,c) = a$ and $(\ell_1,v_1) \xrightarrow{b} (\ell_1',v_1')$ — and thus there is a transition [5] $\ell_1 \xrightarrow{g,b,r} \ell_1'$ such that $v_1 \models g$, $v_1' = v_1[r \leftarrow 0]$ and $v_1' \models \mathit{Inv}(\ell_1')$ — and $(\ell_2,v_2) \xrightarrow{c} (\ell_2',v_2')$ such that $f(b,c) = a$ and $((\ell_1',\ell_2'),v_1'v_2',u) \models \psi$. By induction hypothesis, we have $(\ell_2',v_2',uv_1') \models \psi/\ell_1'$. Then $(\ell_2,v_2,uv_1) \models g \wedge \langle c\rangle\,(r\ \underline{\mathtt{in}}\ \psi/\ell_1')$. Moreover, since $v_1[r \leftarrow 0] \models \mathit{Inv}(\ell_1')$, we have $(\ell_2,v_2,uv_1) \models g \wedge \langle c\rangle\,(r\ \mathtt{in}\ (\mathit{Inv}(\ell_1') \wedge \psi/\ell_1'))$.*

[4] In this table ℓ is a location of $\mathcal{A}_1$ and σ is the synchronisation function.

[5] NB: we assume there exists a transition $\ell_1 \xrightarrow{\mathtt{tt},\bullet,\varnothing} \ell_1$ for each location ℓ_1.

Table II. Quotient construction for two timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$.

$$(\varphi_1 \wedge \varphi_2)/\ell = (\varphi_1/\ell) \wedge (\varphi_2/\ell) \qquad Z/\ell = Z^\ell \text{ where } Z^\ell \text{ is an identifier}$$

$$(\varphi_1 \vee \varphi_2)/\ell = (\varphi_1/\ell) \vee (\varphi_2/\ell)$$

$$\Big(\langle a\rangle\, \varphi\Big)/\ell = \bigvee_{\ell \xrightarrow{g,b,r} \ell' \text{ in } \mathcal{A}_1 \text{ s.t.} \sigma(b,c)=a} g \wedge \langle c\rangle \Big(r \ \underline{\texttt{in}}\ (\mathsf{Inv}(\ell') \wedge \varphi/\ell')\Big)$$

$$\Big([a]\, \varphi\Big)/\ell = \bigwedge_{\ell \xrightarrow{g,b,r} \ell' \text{s.t.} \sigma(b,c)=a} g \Rightarrow [c] \Big(r \ \underline{\texttt{in}}\ (\mathsf{Inv}(\ell') \Rightarrow \varphi/\ell')\Big)$$

$$\Big(\varphi_1\, [\delta\rangle_w\, \varphi_2\Big)/\ell = \Big(\mathsf{Inv}(\ell) \Rightarrow (\varphi_1/\ell)\Big)\, [\delta\rangle_w \Big(\mathsf{Inv}(\ell) \wedge (\varphi_2/\ell)\Big)$$

$$\Big(\varphi_1\, [\delta\rangle_s\, \varphi_2\Big)/\ell = \Big(\mathsf{Inv}(\ell) \Rightarrow (\varphi_1/\ell)\Big)\, [\delta\rangle_s \Big(\mathsf{Inv}(\ell) \wedge (\varphi_2/\ell)\Big)$$

$$(x + c \bowtie y + d)/\ell = (x + c \bowtie y + d) \qquad (x \ \underline{\texttt{in}}\ \varphi)/\ell = x \ \underline{\texttt{in}}\ \Big(\varphi/\ell\Big)$$

Assume $(\ell_2, v_2, uv_1) \models g \wedge \langle c\rangle\ (r\ \underline{\texttt{in}}\ (\mathsf{Inv}(\ell_1') \wedge \psi/\ell_1'))$. *We have* $v_1 \models g$ *and* $v_1[r \leftarrow 0] \models \mathsf{Inv}(\ell_1')$. *Moreover there exists* $(\ell_2, v_2) \xrightarrow{c} (\ell_2', v_2')$ *such that* $(\ell_2', v_2', uv_1[r \leftarrow 0]) \models \psi/\ell_1'$. *By induction hypothesis, we get that* $((\ell_1', \ell_2'), v_1[r \leftarrow 0]v_2', u) \models \psi$. *Clearly we have* $((\ell_1, \ell_2), v_1v_2, u) \models \langle a\rangle\, \varphi$.

– *Assume* $\Psi \stackrel{\text{def}}{=} \varphi_1\, [\delta\rangle_w\, \varphi_2$ *and* $((\ell_1, \ell_2), v_1v_2, u) \models \Psi$. *First assume that we have* $((\ell_1, \ell_2), (v_1v_2) + d, u + d) \models \varphi_1$ *for every* d *such that* $v_1 + d \models \mathsf{Inv}(\ell_1)$ *and* $v_2 + d \models \mathsf{Inv}(\ell_2)$.[6] *By induction hypothesis, we also have* $(\ell_2, v_2 + d, (uv_1) + d) \models \varphi_1/\ell_1$ *for any such* d. *And then we have* $(\ell_2, v_2 + d, (uv_1) + d) \models \mathsf{Inv}(\ell_1) \Rightarrow \varphi_1/\ell_1$ *for any* d *such that* $v_2 + d \models \mathsf{Inv}(\ell_2)$. *Thus* $(\ell_2, v_2, uv_1) \models \Psi/\ell_1$. *Now assume that there exists a* d *such that* $v_1 + d \models \mathsf{Inv}(\ell_1)$, $v_2 + d \models \mathsf{Inv}(\ell_2)$, $((\ell_1, \ell_2), (v_1v_2) + d, u + d) \models \varphi_2$, *and for every* $d' < d$, *we have* $((\ell_1, \ell_2), (v_1v_2) + d', u + d') \models \varphi_1$. *Clearly we also have* $(\ell_2, v_2 + d', (uv_1) + d') \models \mathsf{Inv}(\ell_1) \Rightarrow \varphi_1/\ell_1$ *and* $(\ell_2, v_2 + d, (uv_1) + d) \models \mathsf{Inv}(\ell_1) \wedge \varphi_2/\ell_1$ *and we can deduce* $(\ell_2, v_2, uv_1) \models \Psi/\ell_1$.

Now assume $(\ell_2, v_2, uv_1) \models (\mathsf{Inv}(\ell_1) \Rightarrow \varphi_1/\ell_1)[\delta\rangle_w(\mathsf{Inv}(\ell_1) \wedge \varphi_2/\ell_1)$. *Then, there two cases:*

[6] Note that it implies in particular that it is possible to delay for d units of time from $((\ell_1, \ell_2), v_1v_2)$ in $(\mathcal{A}_1|\mathcal{A}_2)_f$.

- *either: for any delay s.t.* $v_2 + d \models \mathsf{Inv}(\ell_2)$*, we have* $(\ell_2, v_2 + d, uv_1 + d) \models \mathsf{Inv}(\ell_1) \Rightarrow \varphi_1/\ell_1$*, and this entails that after any delay satisfying* $\mathsf{Inv}(\ell_1) \wedge \mathsf{Inv}(\ell_2)$*, we have* φ_1/ℓ_1*. By i.h., we have* $((\ell_1, \ell_2), v_1v_2, u) \models [\delta]\, \varphi_1$ *and then* $((\ell_1, \ell_2), v_1v_2, u) \models [\delta]\, \varphi_1\, [\delta\rangle_w\, \varphi_2$*.*
- *or: there is a delay* d *s.t. (1)* $v_2 + d \models \mathsf{Inv}(\ell_2)$*, (2)* $(\ell_2, v_2 + d, uv_1 + d) \models \mathsf{Inv}(\ell_1) \wedge \varphi_2$
  ell_1 *and (3) for any* $d' < d$*, we have* $(\ell_2, v_2 + d', uv_1 + d') \models \mathsf{Inv}(\ell_1) \Rightarrow \varphi_2/\ell_1$*. As* $v_1 + d \models \mathsf{Inv}(\ell_1)$*, we have* $v_1 + d' \models \mathsf{Inv}(\ell_1)$ *for any* $d' < d$*. By i.h., we can deduce* $((\ell_1, \ell_2), v_1v_2, u) \models \varphi_1\, [\delta\rangle_w\, \varphi_2$*.*

Now consider a formula Φ *with a set of fixpoint variables* $Z_1, \ldots, Z_n$ *and a declaration* $\mathcal{D}$ *assigning a definition* $\mathcal{D}(Z_j)$ *to every* Z_j*. We now consider the unfolding of the variables* Z_j*'s.*

Let $Z_i^{(0)}$ *be the formula* $\mathtt{tt}$ *and, given* $k \in \mathbb{N}$*, let* $Z_i^{(k+1)}$ *be the formula* $\mathcal{D}(Z_i)[Z_j \leftarrow Z_j^{(k)}]_{j=1\ldots n}$*, that is the formula* $\mathcal{D}(Z_i)$ *where any occurrence of the variable* Z_j *is replaced by* $Z_j^{(k)}$ *for* $j = 1, \ldots, n$*. Finally we use* $\Phi^{(k)}$ *to denote the formula* $\Phi[Z_j \leftarrow Z_j^{(k)}]$ *for* $j = 1, \ldots, n$*.*

Thus $\Phi^{(k)}$ *is a variable-free formula and corresponds to the* k*-th unfolding of formula* Φ*. Consider a configuration* s *of a timed automaton, and assume* $s \not\models \Phi$*, this implies that there exists some* k *such that* $s \not\models \Phi^{(k)}$*.*

Moreover one can easily show that $(\Phi^{(k)})/\ell$ *is syntactically similar to* $(\Phi/\ell)^{(k)}$*. From this observation one can prove that the result holds for the full logic. Indeed assume* $((\ell_1, \ell_2), v_1, v_2, u) \not\models \Phi$*, then there exist* k *such that* $((\ell_1, \ell_2), v_1, v_2, u) \not\models \Phi^{(k)}$*. From the first part of the proof, we deduce that we have* $(\ell_2, v_2, uv_1) \not\models (\Phi^{(k)})/\ell_1$*. Thus we have* $(\ell_2, v_2, uv_1) \not\models (\Phi/\ell_1)^{(k)}$ *and then* $(\ell_2, v_2, uv_1) \not\models \Phi/\ell_1$*. The converse is obtained in a similar way.* □

This technique avoids the construction of the exponentially large product corresponding to the parallel composition $(\mathcal{A}_1 | \ldots | \mathcal{A}_n)_\sigma$. However it is fair noticing that this complexity is somehow transferred to the formula: the size of the quotient formula $\Phi/\mathcal{A}_1$ is in $O(|\Phi| \cdot |\mathcal{A}_1|)$. Hence, in order to be used in practice, the compositional method needs to be coupled with reductions (and heuristics therefor): after every quotienting operation, we apply reduction rules in order to keep the size of the formula as small as possible (see [23] for a description of some of these reductions).

The tool CMC (Compositional Model Checker)[7] implements this method for the verification of L_c properties for timed automata. This technique has also been extended to linear hybrid systems [16].

4.3. Characteristic Formulas for Timed Automata

Given a timed automaton $\mathcal{A} = (L, \ell_0, \mathsf{Act}, X, \mathsf{Inv}, T)$, it is possible to build an L_ν formula $\Phi_\mathcal{A}$ that precisely characterizes the behavior of $\mathcal{A}$ in the following sense: a timed automaton will be strongly timed bisimilar to $\mathcal{A}$ iff it satisfies $\Phi_\mathcal{A}$. This construction can be done directly from the definition of $\mathcal{A}$ [24, 3] or it can be seen as a consequence of (1) the ability to express strong timed bisimilarity and (2) L_c be compositional. Indeed consider the variable Z' defined by Equation (*Bisim'*), page 13 to express strong timed bisimilarity; then the formula Z'^{ℓ_0} defined as the quotient[8] Z'/ℓ_0 satisfies the following property for every timed automaton $\mathcal{B}$:

$$\mathcal{A} \cong \mathcal{B} \quad \text{iff} \quad \mathcal{B}' \models Z'^{\ell_0}$$

where $\mathcal{B}'$ is the timed automaton without invariant described in Section 4.1.

Thus, the timed modal logic L_ν (and its superset L_c) is expressive enough to express characteristic formulas for timed automata.

4.4. Comparing L_ν and L_c

Two formulas φ and ψ are *equivalent* over timed automata whenever for every timed automaton $\mathcal{A}$, $\mathcal{A} \models \varphi \;\Leftrightarrow\; \mathcal{A} \models \psi$. A formula φ *can be expressed* in a logic L (over timed automata) whenever there exists a formula $\psi \in L$ equivalent to φ. A logic L is *at least as expressive as* a logic L' whenever every formula φ' can be expressed in L. A logic L is *strictly more expressive than* a logic L' whenever it is at least as expressive as L', and there exists moreover some formula $\varphi \in L$ that cannot be expressed in L'. Two logics L and L' are *equally expressive* (or have the *same expressiveness*) whenever L is at least as expressive as L' and L' is at least as expressive as L.

The operator $[\delta\rangle_w$ has been added to L_ν [13] in order to express controllability (see the next section, where we will see that the modalities $\langle\delta\rangle$ and $[\delta]$ are respectively too weak or too strong for specifying some natural control properties).

The expressiveness gap between L_c and L_ν is for instance witnessed by the classical *CTL*-like formula $\mathsf{E}\varphi\mathsf{W}\psi$, which expresses that "there

[7] *Cf.* `http://www.lsv.ens-cachan.fr/~fl/cmcweb.html`.

[8] With a declaration assigning a definition to every new identifier Z^ℓ.

exists a path along which either φ always holds, or ψ holds at some point and before that point φ holds" (W stands for the weak-until operator). In classical (discrete) Kripke structures, this property can be expressed as the greatest fixpoint of the equation $Y \stackrel{\text{def}}{=} \psi \vee (\varphi \wedge \mathsf{EX}\, Y)$, where EX denotes the next-state operator. In our timed framework, this operator makes no sense because time-elapsing is continuous. And actually, it is the case that the property $\mathsf{E}\varphi\mathsf{W}\psi$ cannot be expressed in L_ν. Indeed from a current state, we need to express that there is a way of letting time elapse such that all visited states satisfy φ, until a state where an action can be performed and so on. The formula $\langle\delta\rangle\,\xi$ allows us to specify that after some delay, the property ξ holds, but there is no requirement on the intermediary states. And the formula $[\delta]\,\xi$ requires that *any* state reachable *via* a delay transition has to satisfy ξ. This is not sufficient to express $\mathsf{E}\varphi\mathsf{W}\psi$, and this is precisely the role of the new modalities $[\delta\rangle_w$ and $[\delta\rangle_s$, as already mentioned in Section 2.

Formally:

THEOREM 6 ([13]). *L_c is strictly more expressive than L_ν over timed automata.*

This result is based on the following technical lemma. Let Φ be the L_c-formula $([a]\,\mathtt{ff})\ [\delta\rangle_w\ (\langle b\rangle\,\mathtt{tt})$.

LEMMA 1 ([13]). *Formula Φ cannot be expressed in L_ν (over timed automata).*

The full proof of this lemma is given in Appendix A, but we give general ideas below.

PROOF 4 (Idea). *The difficulty in the proof is that it is not possible to find two timed automata $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{A} \models \Phi$, $\mathcal{B} \not\models \Phi$ and $\mathcal{A} \models \psi \Leftrightarrow \mathcal{B} \models \psi$ for every $\psi \in L_\nu$. Indeed as we have seen in Section 4.3, L_ν formulas allow us to distinguish between two timed automata that are not bisimilar and if $\mathcal{A} \models \Phi$ and $\mathcal{B} \not\models \Phi$, then $\mathcal{A} \not\cong \mathcal{B}$.*

This is a classical problem in temporal logic [18] where one shows that two temporal logics may have different expressive powers *yet have the same* distinguishing power*. This is why proving that a logic is more expressive than another one can be a rather involved task.*

The general idea of the proof is to build two families of timed automata $(\mathcal{A}_i)_{i\geq 1}$ and $(\mathcal{A}'_i)_{i\geq 1}$ such that for every integer i, $\mathcal{A}_i \models \Phi$ whereas $\mathcal{A}'_i \not\models \Phi$. We then prove that if Φ could be expressed equivalently as formula $\Psi \in L_\nu$ (over timed automata), then there would exist some integer $i \geq 1$ such that $\mathcal{A}'_i \models \Psi$, which is a contradiction. □

5. From Controllability to Model Checking

The main motivation for adding the operator $[\delta\rangle_w$ to L_ν was to express *controllability properties*. The *control problem* asks the following:

"Given a plant $\mathcal{P}$ and a control objective Φ, is there a *controller* $\mathcal{C}$ such that the supervised system $\mathcal{C}(\mathcal{P})$ satisfies Φ?"

The control problem is more general than the verification problem. The plant $\mathcal{P}$ is a (timed) automaton describing the system and its environment, and has two types of actions: controllable actions and uncontrollable actions. The controller can only act on controllable actions in order to restrict the behavior of $\mathcal{P}$ and enforce the property Φ. Moreover the *supervised system* $\mathcal{C}(\mathcal{P})$ can be seen as a simple synchronization function on controllable actions between the two automata (or TTS) that represent $\mathcal{C}$ and $\mathcal{P}$.

In the sequel, we will consider two different kinds of controllers: the *sampling* controllers (performing an action every Δ time units, Δ being fixed), or the more general case of *continuous* controllers that can react in dense-time, the only constraint being that at least Δ time units elapse between two controllable actions.

In the remainder of this section, we show that when the control objective belongs to a *deterministic* fragment of L_c, it is possible to reduce the control problem (the existence of a controller) to a model checking problem. More precisely, given $\mathcal{P}$ a timed automaton and Φ a formula in the deterministic fragment of L_c, the existence of a controller $\mathcal{C}$ such that $\mathcal{C}(\mathcal{P}) \models \Phi$ can be reduced to the model checking problem $(\mathcal{P}|\mathcal{A}_\Delta)_\sigma \models \Phi'$, where $\mathcal{A}_\Delta$ is a simple timed automaton describing the type of the controller (sampling, or continuous), σ synchronizes controllable actions and Φ' is an L_c formula that can be constructed automatically from Φ.

Note that if Φ' holds in $(\mathcal{P}|\mathcal{A}_\Delta)_\sigma$, we know that there exists a controller. However this controller is a TTS and it may be not definable as a timed automaton. Synthesizing a timed automaton $\mathcal{C}$ which is a controller is closely related to the satisfiability problem for L_ν, and the control problem is an open problem when the control objective is in L_ν and $\mathcal{C}$ is required be a timed automaton.

5.1. Plants, Controllers, Controlled Plants

DEFINITION 3 (Plant). *A* plant *is a timed automaton* $\mathcal{P} = (L, \ell_0, \mathsf{Act}, X, \mathit{Inv}, T)$ *such that:*

1. Act *is partitioned into controllable actions (*Act_c*) and uncontrollable actions (*Act_u*);*

2. *it is deterministic w.r.t. every* $a \in \mathsf{Act}_c$*, i.e. for two distinct transitions* $\ell \xrightarrow{g_1,a,r_1} \ell_1$ *and* $\ell \xrightarrow{g_2,a,r_2} \ell_2$ *with* $a \in \mathsf{Act}_c$*, the constraint* $g_1 \wedge g_2$ *is not satisfiable;*

3. *in every state* (ℓ, v) *the timed automaton* $\mathcal{P}$ *can let time elapse or do an uncontrollable action.*

A *controller* [27] for a plant, is a function that during the evolution of the system constantly gives information as to what should be done in order to ensure a control objective Φ. In a given state the controller can either *i*) enforce some particular *controllable* action or *ii*) do nothing at this point, just wait, which will be denoted by the special symbol λ. Of course a controller cannot prevent uncontrollable actions from occurring. Nevertheless, we assume that the controller can **disable** a controllable action at any time, and this will not block the plant due to the point 3 in the definition of a plant

Let $S = (Q, q_0, \mathsf{Act}, \rightarrow_S)$ be a TTS. Given a run $\rho = s_0 \xrightarrow{e_1}_S s_1 \xrightarrow{e_2}_S \cdots \xrightarrow{e_n} s_n \cdots$ in a TTS S (NB: $e_i \in \mathsf{Act} \cup \mathbb{R}_+$), we denote by $\mathit{first}(\rho) = s_0$. If ρ is finite, $\mathit{last}(\rho)$ denotes the last state of ρ. $\mathsf{Runs}(q, S)$ (resp. $\mathsf{Runs}^F(q, S)$) is the set of runs (resp. finite runs) in S starting from state q and $\mathsf{Runs}(S) = \mathsf{Runs}(q_0, S)$ (resp. $\mathsf{Runs}^F(S) = \mathsf{Runs}^F(q_0, S)$) is the set of runs (resp. finite runs) starting from the initial state q_0. We use $s \xrightarrow{e}_S$ as a shorthand for "$\exists s'$ s.t. $s \xrightarrow{e}_S s'$" and extend this notation to finite runs $\rho \xrightarrow{e}_S$ whenever $\mathit{last}(\rho) \xrightarrow{e}_S$. Finally, given a plant $\mathcal{P}$, we use $S_\mathcal{P}$ to denote the TTS corresponding to the semantics of $\mathcal{P}$.

DEFINITION 4 (Controller). *Let* $\mathcal{P} = (L, \ell_0, \mathsf{Act}, X, \mathit{Inv}, T)$ *be a plant. A* controller *over* $\mathcal{P}$ *is a function* f *from* $\mathsf{Runs}^F(S_\mathcal{P})$ *to* $\mathsf{Act}_c \cup \{\lambda\}$ *that satisfies:*[9] $\forall \rho \in \mathsf{Runs}^F(S_\mathcal{P})$*,* $f(\rho) \in \{a_c \mid \rho \xrightarrow{a_c}_{S_\mathcal{P}}\} \cup \{\lambda\}$*.*

The purpose of a controller f for a plant $\mathcal{P}$ is to restrict the set of behaviors in $S_\mathcal{P}$ in order to ensure that some property holds. Supervising (or closing) the plant $\mathcal{P}$ with f produces a set of runs (or TTS) corresponding to the controlled plant, as defined below.

DEFINITION 5 (Controlled plant). *Let* $\mathcal{P} = (L, \ell_0, \mathsf{Act}, X, \mathit{Inv}, T)$ *be a plant,* $q \in S_\mathcal{P}$ *and* f *a controller for* $\mathcal{P}$*. The* controlled plant $f(S_\mathcal{P}, q)$ *is the reachable part of the TTS* $(\mathsf{Runs}^F(q, S_\mathcal{P}), q, \mathsf{Act}, \rightarrow_{\mathcal{P},f,q})$ *defined as follows: if* $\rho \in \mathsf{Runs}^F(S_\mathcal{P}, q)$*, then there will be a transition*

$$\rho \xrightarrow{e}_{\mathcal{P},f,q} (\rho \xrightarrow{e}_\mathcal{P} q')$$

whenever one of the three following conditions is satisfied:

[9] In game theory, a controller is called a *strategy*.

1. $e \in \mathsf{Act}_u$,

2. $e \in \mathsf{Act}_c$ *and* $e = f(\rho)$,

3. $e \in \mathbb{R}_+$ *and for all* $0 \leq e' < e$, $f(\rho \xrightarrow{e'}_{\mathcal{P}}) = \lambda$.

We note $f(\mathcal{P})$ *the plant* $\mathcal{P}$ *controlled by* f *from the initial state of* $\mathcal{P}$.

Note that to define a controller f, it is sufficient to define f for every *reachable* run of $f(\mathcal{P})$ (for unreachable runs, one can always assume that the value returned by f is λ).

5.2. Control Problem

The Δ-dense-time control problem asks for the existence of a controller for a system such that at least $\Delta \geq 0$ time units elapse between two consecutive controllable actions. Such a controller is called a Δ-*controller*, it can prevent time elapsing and enforce a controllable action to happen at any point in time if the time elapsed since the last controllable move is more than Δ time units. If $\Delta = 0$, the Δ-controllers can be arbitrarily fast, they can make two consecutive actions separated by arbitrary small delays (even 0-delay). If $\Delta > 0$, the Δ-controllers are forced to be *strongly non-zeno*. Strong non-zenoness means that two discrete control actions are separated by at least Δ time units. We note $\mathsf{Contr}_\Delta(\mathcal{P})$ the set of Δ-controllers for a plant $\mathcal{P}$.

DEFINITION 6 (Δ-dense-time control problem). *Let* $\mathcal{P} = (L, \ell_0, \mathsf{Act}, X, \mathsf{Inv}, T)$ *be a plant,* $\varphi \in L_c$ *a control objective, and* $\Delta \in \mathbb{Q}_+$. *The* Δ-dense-time control problem *(Δ-CP for short) asks the following:*

$$\textit{Is there a controller } f \in \mathsf{Contr}_\Delta(\mathcal{P}) \textit{ such that } f(\mathcal{P}) \models \varphi? \quad (\Delta\text{-CP})$$

To solve the Δ-dense-time control problem, we consider the self-loop automaton $\mathcal{A}_\Delta$ with transitions labeled by "$z \geq \Delta, a, z := 0$" for every controllable action $a \in \mathsf{Act}_c$. The automaton $\mathcal{A}_\Delta$ is somehow the most general Δ-controller. Given a plant $\mathcal{P}$, we note $\mathcal{P}_\Delta$ the parallel composition $(\mathcal{P}|\mathcal{A}_\Delta)_\sigma$ where σ synchronizes controllable actions (and does not synchronize uncontrollable actions).[10] The new plant $\mathcal{P}_\Delta$ contains all the possible Δ-controllers. Our aim is to reduce the Δ-CP to a model checking problem for $\mathcal{P}_\Delta$, and then, applying the compositionality property of the logic (see section 4.2), to reduce it to a model checking problem for $\mathcal{P}$.

[10] Formally, $\sigma(a, a) = a$ if $a \in \mathsf{Act}_c$ and $\sigma(u, \bullet) = u$ if $u \in \mathsf{Act}_u$.

5.3. L_c^{det}: A Deterministic Fragment of L_c

In the following we will restrict the possible control objectives to properties expressed in a subset L_c^{det} of L_c. Indeed, a control objective of L_c like $\varphi_1 \wedge \varphi_2$ intuitively requires to find a controller that ensures both φ_1 and φ_2. In an inductive construction, this amounts to build a controller that ensures $\varphi_1 \wedge \varphi_2$ from two controllers: one that ensures φ_1 and another one that ensures φ_2. This means that we must be able to merge controllers in a suitable manner. The definition of L_c^{det} will syntactically ensure that the conjunctions of L_c^{det} formulas can be merged safely, *i.e.,* that they are in some sense *deterministic*.

The idea will be to prefix the subformulas φ_is appearing in some conjunction $\varphi_1 \wedge \varphi_2$ with a modal operator involving different actions. Then the existence of a controller for φ_1 and another one for φ_2 will entail the existence of a controller for $\varphi_1 \wedge \varphi_2$.

We first define *basic* terms B_ν by the following grammar:

$$\begin{array}{rl} \alpha \ ::= & \mathtt{tt} \ \mid \ \mathtt{ff} \ \mid \ x \bowtie c \ \mid \ r \ \underline{\mathtt{in}} \ \langle a \rangle \, \varphi \ \mid \ r \ \underline{\mathtt{in}} \ [a] \, \varphi \ \mid \\ & r \ \underline{\mathtt{in}} \ \varphi \, [\delta\rangle_w \, \varphi' \ \mid \ r \ \underline{\mathtt{in}} \ \varphi \, [\delta\rangle_s \, \varphi' \end{array}$$

with $x \in K, r \subseteq K, c \in \mathbb{Q}_+$ and $a \in \mathsf{Act}$ and $\varphi, \varphi' \in L_c^{det}$ (L_c^{det} is defined hereafter). A set of basic terms $A = \{\alpha_1, \alpha_2, \cdots, \alpha_n\}$ is *deterministic* if for all $\sigma \in \mathsf{Act}$ there is at most one i s.t. $\alpha_i = r \ \underline{\mathtt{in}} \ \langle\sigma\rangle\varphi$ or $\alpha_i = r \ \underline{\mathtt{in}} \ [\sigma]\varphi$ and there is at most one i s.t. α_i contains $[\delta\rangle_w$ or $[\delta\rangle_s$. We then define inductively L_c^{det} as the deterministic fragment of L_c as follows:

$$L_c^{det} \ni \ \varphi, \psi ::= \ X \ \mid \ \varphi \vee \psi \ \mid \ \bigwedge_{\alpha \in A} \alpha$$

with $X \in \mathsf{Id}$ and A a (finite) deterministic set of basic terms. With this restriction on the conjunctions, if there are controllers f_α for all $\alpha \in A$, we will be able to merge them to obtain a controller for $\bigwedge_{\alpha \in A} \alpha$.

REMARK 3. *In the untimed case [12] for the μ-calculus control problem, some kind of "deterministic" form is also used when considering satisfiability problems: this is the so-called* disjunctive normal form. *This is not a restriction as all formulas of the μ-calculus can be rewritten in a disjunctive normal form [20]. We do not know yet if this is possible for the timed case, but we conjecture it is not possible in general.*

5.4. From Controllability to Model Checking

In this section, we prove that for any control objective defined as a L_c^{det} formula φ, we can build an L_c formula $\overline{\varphi}$ that holds for $\mathcal{P}_\Delta$ iff there exists a Δ-controller which supervises plant $\mathcal{P}$ in order to satisfy φ.

Let φ be an L_c^{det} formula and $\gamma \in \mathsf{Act}_c \cup \{\lambda\}$. We construct the formula $\overline{\varphi}^{\,\gamma}$ using the inductive translation of Table III. Intuitively, if $a \in \mathsf{Act}_c$, formula $\overline{\varphi}^{\,a}$ will hold when there is a controller which ensures φ and which starts by enforcing controllable action a whereas formula $\overline{\varphi}^{\,\lambda}$ will hold when there is a controller which ensures φ and which starts by delaying. We use the shortcut $\overline{\varphi}$ to express that nothing is required for the strategy, which will correspond to $\bigvee_{\gamma \in \mathsf{Act}_c \cup \{\lambda\}} \overline{\varphi}^{\,\gamma}$. We also use $\langle\lambda\rangle\,\mathtt{tt}$ as a shortcut for $\bigwedge_{a \in \mathsf{Act}_c} [a]\,\mathtt{ff}$.

If we restrict the logic to L_ν^{det} which is the subset of L_c^{det} that does not contain the modalities $[\delta\rangle_w$ and $[\delta\rangle_s$, we still need the operator $[\delta\rangle_w$ to encode the formula $\overline{[\delta]\,\varphi}^{\,\gamma}$. Using the equivalence of equation (3b) page 6, we obtain:

$$\overline{[\delta]\,\varphi}^{\,\gamma} = \overline{\varphi\,[\delta\rangle_w\,\mathtt{ff}}^{\,\gamma} = \begin{cases} \overline{\varphi}^{\,\gamma} & \text{if } \gamma \in \mathsf{Act}_c, \\ \overline{\varphi}^{\,\gamma}\,[\delta\rangle_w \left(\bigvee_{a \in \mathsf{Act}_c} \overline{\varphi}^{\,a} \right) & \text{otherwise} \end{cases} \tag{5}$$

The translation rule introduces the superscript a in the disjunctive right argument of $[\delta\rangle_w$. This just means that we can actually prevent time from elapsing at some point, if we perform a controllable action. On the other hand the formula $\overline{\langle\delta\rangle}^{\,\gamma}$ does not make use of any $[\delta\rangle_w$ or $[\delta\rangle_s$. Using the equivalence of equation (3a) page 6, we obtain:

$$\overline{\langle\delta\rangle\,\varphi}^{\,\gamma} = \overline{\mathtt{tt}\,[\delta\rangle_s\,\varphi}^{\,\gamma} = \begin{cases} \overline{\varphi}^{\,\gamma} & \text{if } \gamma \in \mathsf{Act}_c, \\ \langle\delta\rangle\,\overline{\varphi} & \text{otherwise} \end{cases} \tag{6}$$

because $\overline{\mathtt{tt}}^{\,\lambda}\,[\delta\rangle_s\,\overline{\varphi} = \langle\delta\rangle\,\overline{\varphi}$.

We can now state our main theorem about controllability:

THEOREM 7 ([13]). *Given $\mathcal{P}$ a plant, $\varphi \in L_c^{det}$ a control objective, $\Delta \in \mathbb{Q}_+$, we have:*

$$\Big(\exists f \in \mathit{Contr}_\Delta(\mathcal{P})\ \textit{such that}\ f(\mathcal{P}) \models \varphi\Big) \;\Leftrightarrow\; \mathcal{P}_\Delta \models \overline{\varphi} \tag{7}$$

The proof of Theorem 7 is done by induction on the structure of the formula (see Appendix B).

Table III. Definition of $\overline{\varphi}^{\,\gamma}$, $\varphi \in L_c^{det}$ and $\gamma \in \mathsf{Act}_c \cup \{\lambda\}$

$\overline{\bigwedge_{\alpha\in A} \alpha}^{\,\gamma} \stackrel{\text{def}}{=} \bigwedge_{\alpha\in A} \overline{\alpha}^{\,\gamma}$	$\overline{\bigvee_{\alpha\in A} \alpha}^{\,\gamma} \stackrel{\text{def}}{=} \bigvee_{\alpha\in A} \overline{\alpha}^{\,\gamma}$
$\overline{x \sim c}^{\,\gamma} \stackrel{\text{def}}{=} x \sim c \wedge \langle\gamma\rangle\, \mathtt{tt}$	$\overline{r \ \underline{\mathtt{in}}\ \varphi}^{\,\gamma} \stackrel{\text{def}}{=} r \ \underline{\mathtt{in}}\ \overline{\varphi}^{\,\gamma}$
$\overline{\langle a\rangle\, \varphi}^{\,\gamma} \stackrel{\text{def}}{=} \begin{cases} \mathtt{ff} & \text{if } \gamma, a \in \mathsf{Act}_c \wedge \gamma \neq a \\ \langle a\rangle\, \overline{\varphi} \wedge \langle\gamma\rangle\, \mathtt{tt} & \text{if } a \in \mathsf{Act}_u \\ \langle a\rangle\, \overline{\varphi} & \text{otherwise} \end{cases}$	
$\overline{[a_c]\, \varphi}^{\,\gamma} \stackrel{\text{def}}{=} \begin{cases} \langle\gamma\rangle\, \mathtt{tt} & \text{if } a_c \neq \gamma \\ \langle a_c\rangle\, \overline{\varphi} & \text{if } a_c = \gamma \end{cases}$	$\overline{[a_u]\, \varphi}^{\,\gamma} \stackrel{\text{def}}{=} [a_u]\, \overline{\varphi} \wedge \langle\gamma\rangle\, \mathtt{tt}$
$\overline{\varphi\, [\delta\rangle_w\, \psi}^{\,\gamma} \stackrel{\text{def}}{=} \begin{cases} \overline{\varphi}^{\,\gamma} \vee \overline{\psi}^{\,\gamma} & \text{if } \gamma \in \mathsf{Act}_c \\ \overline{\varphi}^{\,\lambda}\, [\delta\rangle_w \left(\bigvee_{a_c \in \mathsf{Act}_c} (\overline{\varphi}^{\,a_c} \vee \overline{\psi}^{\,a_c}) \right) & \text{if } \gamma = \lambda \end{cases}$	
$\overline{\varphi\, [\delta\rangle_s\, \psi}^{\,\gamma} \stackrel{\text{def}}{=} \begin{cases} \overline{\psi}^{\,\gamma} & \text{if } \gamma \in \mathsf{Act}_c \\ \overline{\varphi}^{\,\lambda}\, [\delta\rangle_s\, \overline{\psi} & \text{if } \gamma = \lambda \end{cases}$	$\overline{X}^{\,\gamma} \stackrel{\text{def}}{=} X_\gamma \wedge \langle\gamma\rangle\, \mathtt{tt}$

This theorem reduces the control problem for properties expressed in L_c^{det} to a model checking problem for properties expressed in L_c. Note however that this theorem does not provide a method to synthesize controllers. We would obtain a synthesis algorithm (due to the compositional property of L_c) if we had a constructive algorithm for solving satisfiability for L_c, but this problem as explained before is still open. Thus we only have results for the synthesis problem under the assumption of bounded resources (clocks and constants) for the controller.

Finally note also that as L_c is compositional, verifying $\mathcal{P}_\Delta \models \overline{\varphi}$ reduces to checking $\mathcal{P} \models \overline{\varphi}\,/\mathcal{A}_\Delta$ where $\mathcal{A}_\Delta$ is the self-loop automaton describing the most permissive controller mentioned before.

5.5. Other Control Problems

We can use the same approach to deal with other kind of control problems, for instance the known-switch condition dense-time control (KSC) [15] or the sampling control.

The KSC problem corresponds to the control of the time-abstract model of a game: intuitively this assumes that time elapsing is not controllable. A controller can thus choose to do a controllable action $a \in \mathsf{Act}_c$ or to do nothing (λ), but in the latter case the controller does not control the duration of the next continuous move.

The *sampling* control problem is a version of the control problem where the controller can perform a controllable action only at dates $k.\Delta$ for $k \in \mathbb{N}$ and $\Delta \in \mathbb{Q}_+$. Δ is the sampling *rate* of the controller.

For these two control problems, we can modify slightly the definition of the $\overline{\varphi}$ and the automaton A_Δ in order to reduce controllability to a model checking problem.

6. Conclusion

In this paper we have presented a set of results about the timed modal logic L_c. The expressiveness results show the ability of this logic to state precisely properties of timed systems. Moreover we have seen that these characteristics allow the use of compositional methods for model checking L_c specifications over timed automata. The control problem for timed automata can also be reduced to a model checking problem when the control objective is expressed in a (deterministic) fragment of L_c. All these results emphasize the interest of timed modal logics for the specification, verification and control of real-time systems.

Acknowledgments. Special thanks to Luca Aceto, Béatrice Bérard and Kim Guldstrand Larsen for all the discussions about timed automata and timed modal logics.

References

1. Aceto, L., P. Bouyer, A. Burgueño, and K. G. Larsen: 1998a, 'The Power of Reachability Testing for Timed Automata'. In: V. Arvind and R. Ramanujam (eds.): *Proceedings of the 18th Conference on Fundations of Software Technology and Theoretical Computer Science (FSTTCS'98)*, Vol. 1530 of *Lecture Notes in Computer Science*. Chennai, India, pp. 245–256.
2. Aceto, L., A. Burgueño, and K. G. Larsen: 1998b, 'Model Checking via Reachability Testing for Timed Automata'. In: *Proc. 4th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS '98), Lisbon, Portugal, Mar. 1998*, Vol. 1384 of *Lecture Notes in Computer Science*. pp. 263–280.
3. Aceto, L., A. Ingólfsdóttir, M. L. Pedersen, and J. Poulsen: 2000, 'Characteristic formulae for timed automata'. *Theoretical Informatics and Applications* **34**(6), 565–584.
4. Aceto, L. and F. Laroussinie: 2002, 'Is Your Model Checker on Time? On the Complexity of Model Checking for Timed Modal Logics'. *Journal of Logic and Algebraic Programming* **52-53**, 7–51.
5. Alur, R., C. Courcoubetis, and D. L. Dill: 1993, 'Model-Checking in Dense Real-Time'. *Information and Computation* **104**(1), 2–34.

6. Alur, R., C. Courcoubetis, and T. A. Henzinger: 1994, 'The observational power of clocks'. In: *Proc. 5th Int. Conf. Theory of Concurrency (CONCUR '94), Uppsala, Sweden, Aug. 1994*, Vol. 836 of *Lecture Notes in Computer Science*. pp. 162–177.
7. Alur, R. and D. L. Dill: 1994, 'A Theory of Timed Automata'. *Theoretical Computer Science* **126**(2), 183–235.
8. Alur, R., T. Feder, and T. A. Henzinger: 1996, 'The benefits of relaxing punctuality'. *Journal of the ACM* **43**(1), 116–146.
9. Alur, R. and T. A. Henzinger: 1992, 'Logics and Models of Real Time: A Survey'. In: *Real-Time: Theory in Practice, Proc. REX Workshop, Mook, NL, June 1991*, Vol. 600 of *Lecture Notes in Computer Science*. pp. 74–106.
10. Alur, R. and T. A. Henzinger: 1994, 'A really temporal logic'. *Journal of the ACM* **41**(1), 181–203.
11. Andersen, H. R.: 1995, 'Partial Model Checking (Extended Abstract)'. In: *Proc. 10th IEEE Symp. Logic in Computer Science (LICS '95), San Diego, CA, USA, June 1995*. pp. 398–407.
12. Arnold, A., A. Vincent, and I. Walukiewicz: 2003, 'Games for synthesis of controllers with partial observation'. *Theoretical Computer Science* **1**(303), 7–34.
13. Bouyer, P., F. Cassez, and F. Laroussinie: 2005, 'Modal Logics for Timed Control'. In: M. Abadi and L. de Alfaro (eds.): *Proceedings of the 16th International Conference on Concurrency Theory (CONCUR'05)*. San Francisco, CA, USA. To appear.
14. Bouyer, P., F. Chevalier, and N. Markey: 2009, 'On the Expressiveness of TPTL and MTL'. *Information and Computation*. To appear.
15. Cassez, F., T. A. Henzinger, and J.-F. Raskin: 2002, 'A Comparison of Control Problems for Timed and Hybrid Systems'. In: *Proc. 5th International Workshop on Hybrid Systems: Computation and Control (HSCC'02)*, Vol. 2289 of *LNCS*. pp. 134–148.
16. Cassez, F. and F. Laroussinie: 2000, 'Model-Checking for Hybrid Systems by Quotienting and Constraints Solving'. In: E. A. Emerson and A. P. Sistla (eds.): *Proceedings of the 12th International Conference on Computer Aided Verification (CAV 2000)*, Vol. 1855 of *Lecture Notes in Computer Science*. Chicago, Illinois, USA, pp. 373–388.
17. Clarke, E. M., O. Grumberg, and D. A. Peled: 1999, *Model Checking*. MIT Press.
18. Emerson, E. A.: 1990, 'Temporal and Modal Logic'. In: J. v. Leeuwen (ed.): *Handbook of Theoretical Computer Science*, Vol. B. Elsevier Science, Chapt. 16, pp. 995–1072.
19. Henzinger, T. A., X. Nicollin, J. Sifakis, and S. Yovine: 1994, 'Symbolic model checking for real-time systems'. *Information and Computation* **111**(2), 193–244.
20. Janin, D. and I. Walukiewicz: 1995, 'Automata for the Modal mu-Calculus and Related Results'. In: *Proc. 20th International Symposium on Mathematical Foundations of Computer Science (MFCS'95)*, Vol. 969 of *Lecture Notes in Computer Science*. pp. 552–562.
21. Laroussinie, F. and K. G. Larsen: 1995a, 'Compositional Model-Checking of Real Time Systems'. In: I. Lee and S. A. Smolka (eds.): *Proceedings of the 6th International Conference on Concurrency Theory (CONCUR'95)*, Vol. 962 of *Lecture Notes in Computer Science*. Philadelphia, Pennsylvania, USA, pp. 529–539.

22. Laroussinie, F. and K. G. Larsen: 1995b, 'Compositional Model Checking of Real Time Systems'. Technical Report RS-95-19, BRICS.
23. Laroussinie, F. and K. G. Larsen: 1998, 'CMC: A Tool for Compositional Model-Checking of Real-Time Systems'. In: S. Budkowski, A. R. Cavalli, and E. Najm (eds.): *Proceedings of IFIP TC6 WG6.1 Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols (FORTE'XI) and Protocol Specification, Testing and Verification (PSTV'XVIII)*, Vol. 135 of *IFIP Conference Proceedings*. Paris, France, pp. 439–456.
24. Laroussinie, F., K. G. Larsen, and C. Weise: 1995, 'From Timed Automata to Logic – and Back'. In: J. Wiedermann and P. Hájek (eds.): *Proceedings of the 20th International Symposium on Mathematical Fundations of Computer Science (MFCS'95)*, Vol. 969 of *Lecture Notes in Computer Science*. Prague, Czech Republic, pp. 27–41.
25. Larsen, K. G., P. Pettersson, and W. Yi: 1995, 'Model-Checking for Real-Time Systems'. In: *Proc. 10th Int. Conf. Fundamentals of Computation Theory (FCT '95), Dresden, Germany, Aug. 1995*, Vol. 965 of *Lecture Notes in Computer Science*. pp. 62–88.
26. Larsen, K. G., P. Pettersson, and W. Yi: 1997, 'UPPAAL in a Nutshell'. *Journal of Software Tools for Technology Transfer* **1**(1–2), 134–152.
27. Maler, O., A. Pnueli, and J. Sifakis: 1995, 'On the Synthesis of Discrete Controllers for Timed Systems'. In: *Proc. 12th Annual Symposium on Theoretical Aspects of Computer Science (STACS'95)*, Vol. 900 of *Lecture Notes in Computer Science*. pp. 229–242.
28. Manna, Z. and A. Pnueli: 1992, *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer.
29. Milner, R.: 1989, 'A Complete Axiomatisation for Observational Congruence of Finite-State Behaviours'. *Information and Computation* **81**(2), 227–247.
30. Ouaknine, J. and J. Worrell: 2005, 'On the decidability of metric temporal logic'. In: *Proc. 20th IEEE Symp. Logic in Computer Science (LICS 2005), Chicago, IL, USA, June 2005*.
31. Park, D.: 1981, 'Concurrency and Automata on Infinite Sequences'. In: *Proc. 5th GI Conf. on Theor. Comp. Sci., Karlsruhe, FRG, Mar. 1981*, Vol. 104 of *Lecture Notes in Computer Science*. pp. 167–183.
32. Yovine, S.: 1997, 'Kronos: A Verification Tool for real-Time Systems'. *Journal of Software Tools for Technology Transfer* **1**(1–2), 123–133.

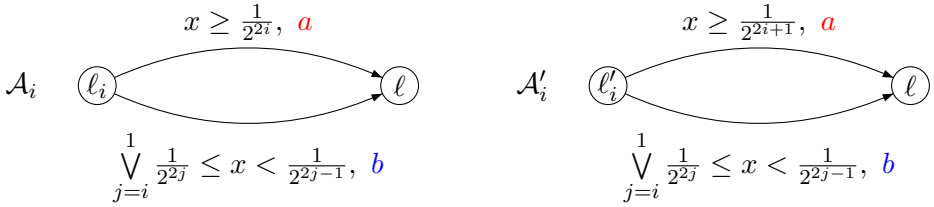
Appendix

A. Proof of Lemma 1

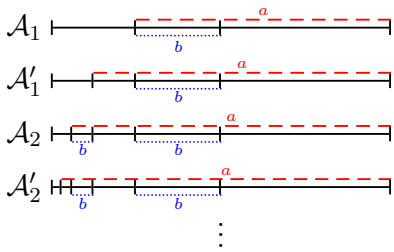
Consider the L_c-formula $\Phi \stackrel{\text{def}}{=} ([a]\,\mathtt{ff})\ [\delta\rangle_w\ (\langle b\rangle\,\mathtt{tt})$, and assume that there exists some L_ν formula Ψ equivalent to Ψ over timed automata.

Of course, a declaration $\mathcal{D}$ may be associated with Ψ to assign an L_ν formula to every identifier in Ψ. In the following, all the transformations we make over Ψ are also done over the formulas occurring in $\mathcal{D}$.

The two families of models. For each $i \geq 1$, we define two timed automata $\mathcal{A}_i$ and $\mathcal{A}'_i$ as follows:



We assume that every location has an invariant $x < 1$. The behaviors of automata $\mathcal{A}_i$ and $\mathcal{A}'_i$ can be represented by (and infered from) the following picture.



It is easy to verify that for each $i \geq 0$,

$$\begin{cases} \mathcal{A}_i \models \Phi \\ \mathcal{A}'_i \not\models \Phi \end{cases}$$

Eliminating constants from formula Ψ and building Ψ_1. Given $p \in \mathbb{Q}_{>0}$, a timed automaton $\mathcal{A}$ (resp. an L_c formula φ), we use $\mathcal{A}_{[p]}$ (resp. $\varphi_{[p]}$) to denote the timed automaton $\mathcal{A}$ (resp. the L_c formula φ) where every constant c occurring in the guard and invariant (resp. in the clocks constraints) is replaced by the product $c \cdot p$.

As Φ is an untimed formula and Ψ is equivalent to Φ, we have the following lemma:

LEMMA 2. *For every $p \in \mathbb{Q}_{>0}$, for every TA $\mathcal{A}$, we have: $\mathcal{A} \models \Psi$ iff $\mathcal{A} \models \Psi_{[p]}$.*

PROOF 5. *As Φ is an untimed formula, we have $\mathcal{A} \models \Phi$ iff $\mathcal{A}_{[\frac{1}{p}]} \models \Phi$ and thus $\mathcal{A}_{[\frac{1}{p}]} \models \Psi$ (by def. of Ψ). Then we deduce that $\mathcal{A} \models \Psi_{[p]}$ (because $\mathcal{A} \models \varphi$ iff $\mathcal{A}_{[p']} \models \varphi_{[p']}$ for any $\mathcal{A}$, φ and p').* □

We thus assume that Ψ has only constants strictly larger than 1 or equal to 0. Now note that time in $\mathcal{A}_i$'s and $\mathcal{A}'_i$'s is bounded by 1,[11] and that we have the following result for 1-bounded timed automata:

LEMMA 3. *Given an L_c formula φ where the constants occurring in the constraints are either 0 or strictly greater than 1, and a 1-bounded timed automaton $\mathcal{A}$, we have: $\mathcal{A} \models \varphi \Leftrightarrow \mathcal{A} \models \varphi'$ where φ' is the formula φ where every atomic constraint $x \bowtie c$ with $c > 0$ is replaced by the truth value of the test $1 \bowtie c$.*

We use the previous result to build a new formula denoted Ψ_1 which does not contain constraints involving positive constants. Note that Ψ_1 is *a priori* not equivalent to Ψ over all timed automata, but it is equivalent to Ψ at least over 1-bounded timed automata, and in particular over all automata $\mathcal{A}_i$'s and $\mathcal{A}'_i$'s.

Formula Ψ_1 can be generated by the following grammar:

$$\begin{aligned}\varphi \ ::= \ & \mathtt{tt} \mid \mathtt{ff} \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid x \ \underline{\mathtt{in}} \ \varphi \ \mid \ x \bowtie 0 \\ & \mid [c]\,\varphi \mid \langle c \rangle \, \varphi \mid [\delta]\,\varphi \mid \langle \delta \rangle \, \varphi \mid X \end{aligned} \tag{8}$$

Eliminating clock subformulas. The only time information in formula Ψ_1 is given by formulas of the form "$x \sim 0$" and "$x \ \underline{\mathtt{in}} \ \cdot$". We want to get rid off this information. For each $C \subseteq K$ and every formula φ generated by the grammar (8), we define inductively new formulas $\{\varphi\}^{(C)}$ as follows:

$$\begin{aligned}
\{\alpha\}^{(C)} &= \alpha \quad \text{if } \alpha \in \{\mathtt{tt}, \mathtt{ff}\} \\
\{\varphi_1 \mathit{op} \, \varphi_2\}^{(C)} &= \{\varphi_1\}^{(C)} \mathit{op} \, \{\varphi_2\}^{(C)} \text{ if } \mathit{op} \in \{\wedge, \vee\} \\
\{x \ \underline{\mathtt{in}} \ \varphi\}^{(C)} &= x \ \underline{\mathtt{in}} \ \{\varphi\}^{(C \cup \{x\})} \\
\{[c]\,\varphi\}^{(C)} &= [c]\,\{\varphi\}^{(C)} \\
\{\langle c \rangle \, \varphi\}^{(C)} &= \langle c \rangle \, \{\varphi\}^{(C)} \\
\{[\delta]\,\varphi\}^{(C)} &= \{\varphi\}^{(C)} \wedge [\delta]^+ \, \{\varphi\}^{(\varnothing)} \\
\{\langle \delta \rangle \, \varphi\}^{(C)} &= \{\varphi\}^{(C)} \wedge \langle \delta \rangle^+ \, \{\varphi\}^{(\varnothing)} \\
\{X\}^{(C)} &= X_C \\
\{x > 0\}^{(C)} &= \begin{cases} \mathtt{tt} & \text{if } x \notin C \\ \mathtt{ff} & \text{if } x \in C \end{cases} \\
\{x = 0\}^{(C)} &= \begin{cases} \mathtt{tt} & \text{if } x \in C \\ \mathtt{ff} & \text{if } x \notin C \end{cases}
\end{aligned}$$

where the modality $\langle \delta \rangle^+$ (resp. $[\delta]^+$) corresponds the existential (resp. universal) quantification over positive (and not as previously nonnegative) delay.

[11] We call such an automaton 1-*bounded*.

Intuitively the formula $\{\varphi\}^{(C)}$ is a simplification of φ when we assume that the value of every clock in C is 0 and the value of every clock in $K \backslash C$ is positive.

LEMMA 4. *For every timed automaton $\mathcal{A}$, for every formula φ generated by the grammar (8), for every extended configuration (ℓ, uv) of $\mathcal{A}$, we have:*

$$(\ell, uv) \models \varphi \;\Leftrightarrow\; (\ell, uv) \models \{\varphi\}^{(C)}$$

where $C = \{x \in K \mid v(x) = 0\}$. In particular, in the initial configuration (all clocks set to 0), we have:

$$(\ell_0, u_0 v_0) \models \varphi \;\Leftrightarrow\; (\ell_0, u_0 v_0) \models \{\varphi\}^{(K)}$$

The new formula $\{\Psi_1\}^{(K)}$ has no more clock constraints. We can thus erase all operators "x $\underline{\texttt{in}}$ $\cdot$" because clocks are no more used. We get a new formula Ψ_2 (without clocks) which is generated by the grammar

$$\varphi ::= \texttt{tt} \mid \texttt{ff} \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid [c]\,\varphi \mid \langle c \rangle\, \varphi \mid [\delta]^{+}\, \varphi \mid \langle \delta \rangle^{+}\, \varphi \mid X \quad (9)$$

and such that Ψ_2 is equivalent to Φ over 1-bounded timed automata.

Region abstraction. The regions[12] for automaton $\mathcal{A}_i$ in state ℓ_i correspond to the following set of intervals (denoted $\mathcal{R}_i$) :

$$\left[0, \frac{1}{2^{2i}}\right[,\ \left[\frac{1}{2^{2i}}, \frac{1}{2^{2i-1}}\right[,\ \cdots,\ \left[\frac{1}{2}, 1\right[$$

whereas the regions for automaton $\mathcal{A}'_i$ in state ℓ'_i are based on the following set of intervals (denoted $\mathcal{R}'_i$):

$$\left[0, \frac{1}{2^{2i+1}}\right[,\ \left[\frac{1}{2^{2i+1}}, \frac{1}{2^{2i}}\right[,\ \cdots,\ \left[\frac{1}{2}, 1\right[$$

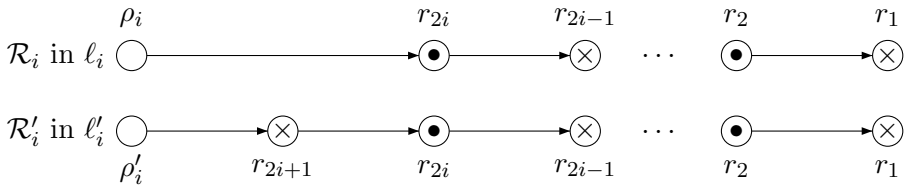
We use r_k, ρ_k and ρ'_k to denote the following intervals:

$$\begin{cases} r_k = \left[\dfrac{1}{2^k}, \dfrac{1}{2^{k-1}}\right[\\ \rho_k = \left[0, \dfrac{1}{2^{2k}}\right[\\ \rho'_k = \left[0, \dfrac{1}{2^{2k+1}}\right[\end{cases}$$

[12] Following the definition in [7].

An *immediate successor (by time elapsing) operator* Succ can be defined in a natural way: $\mathsf{Succ}(\ell_i, \rho_i) = (\ell_i, r_{2i})$, $\mathsf{Succ}(\ell_i, r_{2i}) = (\ell_i, r_{2i-1})$, ... and $\mathsf{Succ}(\ell_i, r_1)$ is undefined.

Moreover note that for these automata, only one region is useful in state ℓ, namely $(\ell, [0, 1[)$. The interesting parts of these automata are thus locations ℓ_i and ℓ'_i because no action nor delays increasing clock x above 1 are allowed from ℓ. We focus on those two locations, and first notice that all regions are right-open. In the following figure, we represent the time evolution and part of the region automaton in locations ℓ_i and ℓ'_i.



In the previous figure, the labelling "•" means that both a and b can be done (leading to state ℓ) whereas the labelling "×" means that only action a can be done (also leading to state ℓ). Finally no labelling means that no action can be done. We will notice something stronger in that states (ℓ_i, r_j) and (ℓ'_i, r_j) (when $j \geq 2i$) do abstract states that satisfy the same formulas generated by grammar (9). More precisely:

LEMMA 5. *Let ψ be a formula generated by grammar (9) and $j \geq 1$.*

- *Given two valuations u and v in r_j, for every i such that $2i \geq j$ and for every k such that $2k + 1 \geq j$, we have:*

$$(\ell_i, u) \models \psi \;\Leftrightarrow\; (\ell_i, v) \models \psi \;\Leftrightarrow\; (\ell'_k, u) \models \psi \;\Leftrightarrow\; (\ell'_k, v) \models \psi$$

- *For all valuations u and v in ρ_j, we have:*

$$(\ell_j, u) \models \psi \;\Leftrightarrow\; (\ell_j, v) \models \psi$$

- *For all valuations u and v in ρ'_j, we have:*

$$(\ell'_j, u) \models \psi \;\Leftrightarrow\; (\ell'_j, v) \models \psi$$

The proof is a direct consequence of the properties of the region abstraction.

Atomic propositions. Now we are going to replace every subformulas $[c]\,\varphi$ or $\langle c\rangle\,\varphi$ with $c \in \mathsf{Act}$ by $\mathtt{tt}$, $\mathtt{ff}$, $\langle c\rangle\,\mathtt{tt}$ or $[c]\,\mathtt{ff}$.

If $c \notin \{a, b\}$, then $\langle c \rangle\, \varphi$ (resp. $[c]\, \varphi$) is clearly equivalent to $\mathtt{ff}$ (resp. $\mathtt{tt}$) when it is interpreted over a state of $\mathcal{A}_i$ or $\mathcal{A}'_i$.

Now assume $c \in \{a, b\}$ and consider a formula ψ of the form $\langle c \rangle\, \varphi$ or $[c]\, \varphi$. As there is only one region in ℓ, we have:

$$[\![\varphi]\!] \cap \{(\ell, u) \mid u \in [0, 1[\} = \begin{cases} \{(\ell, u) \mid u \in [0, 1[\} \\ \text{or } \varnothing \end{cases}$$

Thus φ is equivalent to $\mathtt{tt}$ or $\mathtt{ff}$ over the region associated with ℓ, and this Boolean value depends only on ℓ and not on the region of ℓ_i or ℓ'_i where ψ is interpreted. Thus we can replace $[c]\, \varphi$ (resp. $\langle c \rangle\, \varphi$) by $\mathtt{tt}$[13] or $[c]\, \mathtt{ff}$ (resp. $\langle e \rangle\, \mathtt{tt}$ or $\mathtt{ff}$) depending on the truth value of φ in ℓ.

From these remarks we can define a new formula Ψ_3 equivalent to Ψ_2 over the states of $\mathcal{A}_i$ and $\mathcal{A}'_i$ (*i.e.,* $\forall u \in [0; 1[$, we have $(\ell_i, u) \models \Psi_2$ iff $(\ell'_i, u) \models \Psi_3$) and generated by the following grammar (with $c \in \{a, b\}$):

$$\varphi ::= \mathtt{tt} \mid \mathtt{ff} \mid [c]\mathtt{ff} \mid \langle c \rangle \mathtt{tt} \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid [\delta]^+ \varphi \mid \langle \delta \rangle^+ \varphi \mid X \quad (10)$$

Untiming the formula. We will now build a new formula Ψ_4 using new modalities (G^+ and F^+), which will be interpreted over regions of $\mathcal{A}_i$ and $\mathcal{A}'_i$. Furthermore Ψ_3 and Ψ_4 will be equivalent over $\mathcal{A}_i$ and $\mathcal{A}'_i$ in the following sense: for every $i \geq 1$, we will have that $(\ell_i, \rho_i) \models \Psi_4$ iff $(\ell_i, u_0) \models \Psi_3$, and $(\ell'_i, \rho'_i) \models \Psi_4$ iff $(\ell'_i, u_0) \models \Psi_3$, where u_0 is the valuation setting each clock to 0.

We define a new logic by the grammar:

$$\varphi ::= \mathtt{tt} \mid \mathtt{ff} \mid [c]\, \mathtt{ff} \mid \langle c \rangle\, \mathtt{tt} \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{G}^+ \varphi \mid \mathsf{F}^+ \varphi \mid X \quad (11)$$

This logic is interpreted over states of the region automaton (*i.e.,* pairs (l, r) where l is a location of a timed automaton and r is a region). The semantics (denoted $\vdash$) corresponds to the standard symbolic semantics of L_ν over regions (see [24]) except for the new modalities, which is defined as follows:

$$(l, r) \vdash \mathsf{F}^+ \varphi \overset{\text{def}}{\Leftrightarrow} \exists r' \in \mathsf{Succ}^+(r) \text{ such that } (l, r') \vdash \varphi$$
$$(l, r) \vdash \mathsf{G}^+ \varphi \overset{\text{def}}{\Leftrightarrow} \forall r' \in \mathsf{Succ}^+(r) \text{ we have } (l, r') \vdash \varphi$$

Note that as every interval of $\mathcal{R}_i$ and $\mathcal{R}'_i$ is right-open, when $[\delta]^+ \psi$ holds at a state within a region (ℓ_i, r) or (ℓ'_i, r), this region has to contain valuations satisfying ψ, and this makes a difference with the new modality G^+ that only deals with regions in the future. The same difference occurs for $\langle \delta \rangle^+$ and F^+. Then we define Ψ_4 as the formula

[13] Note that $[c]\, \mathtt{tt} \equiv \mathtt{tt}$.

Ψ_3 where every subformula $[\delta]^+ \psi$ is replaced by $\psi \wedge \mathsf{G}^+\psi$ and every subformula $\langle\delta\rangle^+ \psi$ is replaced by $\psi \vee \mathsf{F}^+\psi$. It is then easy to prove the following lemma:

LEMMA 6. *Given $i \geq 1$, for every region $r \in \mathcal{R}_i$ and for every $u \in r$,*

$$(\ell_i, u) \models \Psi_3 \;\Leftrightarrow\; (\ell_i, r) \vdash \Psi_4.$$

For every $r \in \mathcal{R}'_i$ and for every $u \in r$,

$$(\ell'_i, u) \models \Psi_3 \;\Leftrightarrow\; (\ell'_i, r) \vdash \Psi_4.$$

The new formula Ψ_4 is now an untimed formula interpreted over a discrete structure (the region automaton). Moreover, Ψ_4 is equivalent to Ψ_3 (and hence to Φ) over $\mathcal{A}_i$ and $\mathcal{A}'_i$.

Gluing everything. The formula Ψ_4 (and its declaration $\mathcal{D}_4$) can be written in normal form as a system of equations $(X_i = f_i(X_1, ..., X_n))_{1 \leq i \leq n}$ with $\Psi_4 = X_1$. We assume that each formula $f_i(X_1, ..., X_n)$ is a Boolean combination b_i of subformulas α_i^j (which can be either some formula $\mathsf{F}^+\beta_i^j$, or $\mathsf{G}^+\beta_i^j$, or some atomic-like formula $\langle a\rangle\,\mathtt{tt}$, $[a]\,\mathtt{ff}$, $\mathtt{tt}$ or $\mathtt{ff}$, or some fix-point variable X_i^j):

$$\begin{cases} X_1 \stackrel{\text{def}}{=} b_1(\alpha_1^1, ..., \alpha_1^{k_1}) \\ \quad\vdots \\ X_n \stackrel{\text{def}}{=} b_n(\alpha_n^1, ..., \alpha_n^{k_n}) \end{cases}$$

Without loss of generality we can assume that no subformula α_i^j is a fix-point variable X_k. Indeed assume that we have $\alpha_i^j = X_k$ (with $i \neq k$), then the new formula obtained by replacing X_k by its definition formula is equivalent to the previous formula. And if we have $\alpha_i^j = X_i$, then the new formula obtained by replacing this variable X_i by $\mathtt{tt}$ is equivalent to the initial formula. Thus, each α_i^j is either a "basic" formula $\mathtt{tt}$, $\mathtt{ff}$, $\langle c\rangle$, $[c]\,\mathtt{ff}$ or X, or a formula $\mathsf{F}^+\varphi$ or a formula $\mathsf{G}\varphi$.

From the structure of the region automata of $\mathcal{A}_i$ and $\mathcal{A}'_i$, we have the following lemma:

LEMMA 7. *Let φ be a formula generated by the grammar (11). For every $i \geq 1$, we have the following implications:*

- $(\ell_i, \rho_i) \vdash \mathsf{F}^+\varphi$ *implies* $(\ell'_i, \rho'_i) \vdash \mathsf{F}^+\varphi$
- $(\ell_i, \rho_i) \not\vdash \mathsf{G}^+\varphi$ *implies* $(\ell'_i, \rho'_i) \not\vdash \mathsf{G}^+\varphi$
- $(\ell_i, \rho_i) \vdash \mathsf{G}^+\varphi$ *implies* $(\ell'_{i-1}, \rho'_{i-1}) \vdash \mathsf{G}^+\varphi$

– $(\ell_i, \rho_i) \not\vdash \mathsf{F}^+\varphi$ *implies* $(\ell'_{i-1}, \rho'_{i-1}) \not\vdash \mathsf{F}^+\varphi$

– $(\ell'_i, \rho'_i) \vdash \mathsf{F}^+\varphi$ *implies* $(\ell'_{i+1}, \rho'_{i+1}) \vdash \mathsf{F}^+\varphi$

We are now in a position where we can finish the proof and show that Ψ_4 as previously described (and hence Ψ) cannot exist.

As X_1 holds at any (ℓ_i, ρ_i) (because Ψ_4 is equivalent to Φ over $\mathcal{A}_i$), there exists a sequence $(\gamma_j)_{1\leq j\leq k_1} \in \{\mathtt{tt}, \mathtt{ff}\}^{k_1}$ such that (1) $b_1(\alpha_1^1 \leftarrow \gamma_1, ..., \alpha_1^{k_1} \leftarrow \gamma_{k_1})$ is true, and (2) there is an infinite sequence of indexes I such that for every $i \in I$, for every $1 \leq j \leq k_1$, $\gamma_j = \mathtt{tt}$ iff $(\ell_i, \rho_i) \models \alpha_1^j$.[14]

Let α_{F} be the set $\{\alpha_1^i \mid \gamma_i = \mathtt{tt} \text{ and } \alpha_1^i = \mathsf{F}^+\beta_1^i\}$, $\alpha_{\neg\mathsf{F}}$ be the set $\{\alpha_1^i \mid \gamma_i = \mathtt{ff} \text{ and } \alpha_1^i = \mathsf{F}^+\beta_1^i\}$, α_{G} be the set $\{\alpha_1^i \mid \gamma_i = \mathtt{tt} \text{ and } \alpha_1^i = \mathsf{G}^+\beta_1^i\}$, and $\alpha_{\neg\mathsf{G}}$ be the set $\{\alpha_1^i \mid \gamma_i = \mathtt{ff} \text{ and } \alpha_1^i = \mathsf{G}^+\beta_1^i\}$.

From the two first implications of Lemma 7, for every $i \in I$, $(\ell'_i, \rho'_i) \models \bigwedge \alpha_{\mathsf{F}} \wedge \bigwedge \alpha_{\neg\mathsf{G}}$. Given $i_0 \in I$, we also have that for every $i \geq i_0$, $(\ell'_i, \rho'_i) \models \bigwedge \alpha_{\mathsf{F}} \wedge \bigwedge \alpha_{\neg\mathsf{G}}$ (because the future of $(\ell'_{i+1}, \rho'_{i+1})$ contains the future of (ℓ'_i, ρ'_i)). From the third and fourth implications, we get that for every $i \in I$, we have that $(\ell'_{i-1}, \rho'_{i-1}) \models \bigwedge \alpha_{\mathsf{G}} \wedge \bigwedge \alpha_{\neg\mathsf{F}}$. Thus we can easily find some i such that $(\ell'_i, \rho'_i) \models \bigwedge \alpha_{\mathsf{F}} \wedge \bigwedge \alpha_{\neg\mathsf{G}} \wedge \bigwedge \alpha_{\mathsf{G}} \wedge \bigwedge \alpha_{\neg\mathsf{F}}$ and thus $(\ell'_i, \rho'_i) \models \Psi_4$ (because the other "basic" subformulas can be easily treated) and then $(\ell'_i, \rho'_i) \models \Phi$ which contradicts the fact that $\mathcal{A}'_i \not\models \Phi$.

We conclude that there is no formula Ψ in L_ν which is equivalent to the formula $\Phi = ([a]\,\mathtt{ff})\ [\delta\rangle_w\ (\langle b\rangle\,\mathtt{tt})$. □

B. Proof of Theorem 7

Before proving Theorem 7 we introduce some new notations.

Let $S = (Q, q_0, \mathsf{Act}, \longrightarrow_S)$ be an *acyclic* TTS. Here we assume that S is an unfolding of a TTS. This assumption could also be done – with no change – to define the semantics of the TAs or to interpret the L_c formulas: this point of view corresponds to the (infinite) execution tree of the system. And this assumption allows us to see $f(S, q)$ as a sub-TTS (*i.e.,* sub-tree) of S for any controller f.

For $q \in Q$ and $\Delta \in \mathbb{Q}_+$ we define:

– S^q_Δ to be the sub-TTS of S rooted at q such that two consecutive controllable actions (in Act_c) are separated by a time amount t such that $t \geq \Delta$;

– S_Δ stands for $S^{q_0}_\Delta$;

[14] There is a way to witness X_1 be true which appears infinitely often.

- for $\tau \leq \Delta$, $S^q_{\Delta,\tau}$ is the sub-TTS of S^q_Δ where no controllable action occurs before τ time units from the root q;
- $\mathsf{Contr}(S^q_{\Delta,\tau}, q)$ is the set of controllers for TTS $S^q_{\Delta,\tau}$ from state q; $\mathsf{Contr}(S^{q_0}_{\Delta,0}, q_0)$ thus denotes the set of controllers that can let at least Δ time units between two consecutive controllable actions.

If S is the semantics of a plant $\mathcal{P} = (L, \ell_0, \mathsf{Act}, X, \mathsf{Inv}, T)$, the TTS $S^{q_0}_\Delta$ can be effectively constructed using a parallel composition with a self-loop automaton $\mathcal{A}_\Delta$ with a fresh clock x enforcing a delay larger than or equal to Δ (e.g. by $x \geq \Delta$) between two controllable actions. We denote $\mathcal{P}_\Delta$ the synchronized product $(\mathcal{P}|\mathcal{A}_\Delta)_\sigma$ where $\sigma(a,a) = a$ if $a \in \mathsf{Act}_c$ and $\sigma(u, \bullet) = u$ if $u \in \mathsf{Act}_u$.

Theorem 7 will be a consequence of the following lemma:

LEMMA 8. *For any $\gamma \in \mathsf{Act}_c \cup \{\lambda\}$, any state $q \in Q$ and any L^{det}_c formula Φ, we have:*

$$\Big(\exists f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)\ s.t.\ f(S,q), (q,v) \models \Phi \wedge \langle\gamma\rangle\, \mathtt{tt}\Big) \Leftrightarrow S^q_{\Delta,\tau}, (q,v) \models \overline{\Phi}^{\,\gamma}$$

Note how this lemma interprets for formulas $\overline{\Phi}$:

$$\Big(\exists f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q) \text{ s.t. } f(S,q), (q,v) \models \Phi\Big) \Leftrightarrow \Big(S^q_{\Delta,\tau}, (q,v) \models \overline{\Phi}\Big)$$

PROOF 6. *First we assume that the result holds for the fixpoint variables and we show the lemma by structural induction over L^{det}_c formulas. The cases $\Phi \stackrel{\text{def}}{=} x \sim c$ or $\Phi \stackrel{\text{def}}{=} r\ \underline{\mathtt{in}}\ \varphi$ are obvious. We prove the lemma for the discrete modalities $[a_u]$, $\langle a_u\rangle$, $\langle a_c\rangle$ and $[a_c]$ (assuming $a_c \in \mathsf{Act}_c$ and $a_u \in \mathsf{Act}_u$). Because of the dualities of the operators $[\delta]$, $\langle\delta\rangle$, $[\delta\rangle_w$ and $[\delta\rangle_s$ (equations (3a)–(4b)), we only prove the lemma on a complete pair of timed operators, namely for $[\delta\rangle_s$ and $[\delta]$.*

- $\Phi \stackrel{\text{def}}{=} [a_u]\,\varphi$*:*
  - $\Rightarrow$ *Assume there exists $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ such that $f(S,q), (q,v) \models [a_u]\,\varphi \wedge \langle\gamma\rangle\,\mathtt{tt}$. Then for any $q \xrightarrow{a_u} q'$ in $f(S,q)$, we have $f(S,q), (q',v) \models \varphi$. Then there exists $f' \in \mathsf{Contr}(S^q_{\Delta,\tau}, q')$ such that $f'(S,q'), (q',v) \models \varphi$ and the induction hypothesis provides: $S^{q'}_{\Delta,\tau}, (q',v) \models \overline{\varphi}$. Since the strategies cannot block the uncontrollable actions, any action $a_u \in \mathsf{Act}_u$ that can be performed from (q,v) in $S^q_{\Delta,\tau}$, can also be performed in $f(S,q)$ and then $S^q_{\Delta,\tau}, (q,v) \models [a_u]\,\overline{\varphi}$. Moreover $f(S,q), (q,v) \models \langle\gamma\rangle\,\mathtt{tt}$ which implies that $S^q_{\Delta,\tau}, (q,v) \models \langle\gamma\rangle\,\mathtt{tt}$, and thus $S^q_{\Delta,\tau}, (q,v) \models \overline{[a_u]\,\varphi}^{\,\gamma}$.*

$\Leftarrow$ *Assume* $S^q_{\Delta,\tau}, (q,v) \models [a_u]\,\overline{\varphi} \wedge \langle\gamma\rangle\,\mathtt{tt}$. *For any transition* $S^q_{\Delta,\tau}, q \xrightarrow{a_u} s'$ *with* $a_u \in \mathsf{Act}_u$, *we have* $S^q_{\Delta,\tau}, (q',v) \models \overline{\varphi}$. *By induction hypothesis, we know that there exists* $f_{a_u} \in \mathsf{Contr}(S^q_{\Delta,\tau}, q')$ *such that* $f_{a_u}(S,q'), (q',v) \models \varphi$. *Let* f *be the strategy defined by:* $f(q \xrightarrow{a_u} \rho) \stackrel{\text{def}}{=} f_{a_u}(\rho)$ *for any* ρ *starting in state* q' *and* $f(q) = \gamma$ *if* $\gamma \in \mathsf{Act}_c$ *(note that in that case, it is possible to do a* γ *because* $S^q_{\Delta,\tau}, (q,v) \models \langle\gamma\rangle\mathtt{tt}$*), or* $f(q) = \lambda$ *otherwise.*

– $\Phi \stackrel{\text{def}}{=} \langle a_u\rangle\,\varphi$*:*

$\Rightarrow$ *Assume there exists* $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *such that* $f(S,q), (q,v) \models \langle a_u\rangle\,\varphi \wedge \langle\gamma\rangle\,\mathtt{tt}$. *Then there exists* $s \xrightarrow{a_u} s'$ *in* $f(S,q)$ *with* $f(S,q), (q',v) \models \varphi$. *Therefore there is* $f' \in \mathsf{Contr}(S^q_{\Delta,\tau}, q')$ *such that* $f'(S,q'), (q',v) \models \varphi$ *and the induction hypothesis entails:* $S^{q'}_{\Delta,\tau}, (q',v) \models \overline{\varphi}$. $S^q_{\Delta,\tau}$ *contains the behaviors of* $f(S,q)$, *then* $S^q_{\Delta,\tau}, (q,v) \models \langle a_u\rangle\,\overline{\varphi}$. *Moreover,* $f(S,q), (q,v) \models \langle\gamma\rangle\,\mathtt{tt}$, *thus* $S^q_{\Delta,\tau}, (q,v) \models \langle a_u\rangle\,\overline{\varphi} \wedge \langle\gamma\rangle\,\mathtt{tt}$, *and thus* $S^q_{\Delta,\tau}, (q,v) \models \overline{\langle a_u\rangle\,\varphi}^{\,\gamma}$.

$\Leftarrow$ *Assume* $S^q_{\Delta,\tau}, (q,v) \models \langle a_u\rangle\,\overline{\varphi} \wedge \langle\gamma\rangle\,\mathtt{tt}$. *There is a transition* $S^q_{\Delta,\tau}, q \xrightarrow{a_u} q'$ *such that* $S^q_{\Delta,\tau}, (q',v) \models \overline{\varphi}$. *By induction hypothesis, we know that there exists* $f_{a_u} \in \mathsf{Contr}(S^q_{\Delta,\tau}, q')$ *such that* $f_{a_u}(S,q'), (q',v) \models \varphi$. *Let* f *be the strategy defined by:* $f(q \xrightarrow{a_u} \rho) \stackrel{\text{def}}{=} f_{a_u}(\rho)$ *for any* ρ *starting in state* q' *and* $f(q) = \gamma$ *if* $\gamma \in \mathsf{Act}_c$, *or* $f(q) = \lambda$ *otherwise.*

– $\Phi \stackrel{\text{def}}{=} \langle a_c\rangle\,\varphi$*:*

$\Rightarrow$ *There exists* $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *such that* $f(S,q), (q,v) \models \langle a_c\rangle\,\varphi \wedge \langle\gamma\rangle\,\mathtt{tt}$. *Then clearly* γ *is* a_c*: otherwise this would entail that* f *is not deterministic and requires two different controllable actions from the state* (q,v). *There exists* $f(S,q) : q \xrightarrow{a_c} q'$ *such that* $f(S,q'), (q',v) \models \varphi$. *Moreover defining* $f' \in \mathsf{Contr}(S^q_{\Delta,\Delta}, q')$ *by* $f'(\rho) \stackrel{\text{def}}{=} f(q \xrightarrow{a_c} \rho)$ *for any* ρ *starting in* q', *we get that* $f'(S,q'), (q',v) \models \varphi$. *By induction hypothesis we have* $S^{q'}_{\Delta,\Delta}, (q',v) \models \overline{\varphi}$. $S^q_{\Delta,\tau}$ *contains the behaviors of* $f(S,q)$, *then* $S^q_{\Delta,\tau}, (q,v) \models \langle a_c\rangle\,\overline{\varphi}$ *and thus* $S^q_{\Delta,\tau}, (q,v) \models \overline{\langle a_c\rangle\,\varphi}^{\,\gamma}$.

$\Leftarrow$ *The only possible case is* $\gamma = a_c$ *and* $S^q_{\Delta,\tau}, (q,v) \models \langle a_c\rangle\,\overline{\varphi}$. *There is a transition* $S^q_{\Delta,\tau}, q \xrightarrow{a_c} s'$ *such that* $S^{q'}_{\Delta,\Delta}, (q',v) \models$

$\overline{\varphi}$ *. By induction hypothesis we know that there exists* $f' \in \mathsf{Contr}(S^q_{\Delta,\Delta}, q')$ *such that* $f'(S,q'),(q',v) \models \varphi$*. Let* f *be the strategy defined by:* $f(q \xrightarrow{a_c} \rho) \stackrel{\text{def}}{=} f'(\rho)$ *for any* ρ *run starting in* q' *and* $f(q) = a_c$*.* f *is a* Δ*-strategy and belongs to* $\mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *— note that in this case* $\tau = 0$ *—, and* $f(S,q),(q,v) \models \langle a_c \rangle\, \varphi$ *and then* $\langle a_c \rangle\, \mathtt{tt}$ *also holds for* $f(S,q),(q,v)$*.*

– $\Phi \stackrel{\text{def}}{=} [a_c]\, \varphi$*:*

⇒ *If* $a_c \neq \gamma$ *the result is obvious. Now assume* $a_c = \gamma$*, then there exists* $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *such that* $f(S,q),(q,v) \models [a_c]\, \varphi \wedge \langle a_c \rangle\, \mathtt{tt}$*. The same proof as above (for* $\Phi \stackrel{\text{def}}{=} \langle a_c \rangle\, \varphi$*) gives* $S^q_{\Delta,\tau},(q,v) \models \langle a_c \rangle\, \overline{\varphi}$ *, i.e.* $S^q_{\Delta,\tau},(q,v) \models \overline{[a_c]\, \varphi}^{\gamma}$ *because* $S^q_{\Delta,\tau}$ *is deterministic.*

⇐ *First assume* $\gamma \in \mathsf{Act}_c \setminus \{a_c\}$ *or* $\gamma = \lambda$*. Then* $S^q_{\Delta,\tau},(q,v) \models \langle \gamma \rangle\, \mathtt{tt}$ *we define the strategy* f *to be* $f(q) = \gamma$*. This allows us to have* $f(S,q),(q,v) \models [a_c]\, \varphi \wedge \langle \gamma \rangle\, \mathtt{tt}$ *(as* a_c *is disabled by* f*). Finally assume* $\gamma = a_c$*. Then we have* $S^q_{\Delta,\tau},(q,v) \models \langle a_c \rangle\, \overline{\varphi}$ *. There exists* $S^q_{\Delta,\tau}, q \xrightarrow{a_c} s'$ *such that* $S^{q'}_{\Delta,\Delta},(q',v) \models \overline{\varphi}$ *. By induction hypothesis, there exists* $f \in \mathsf{Contr}(S^q_{\Delta,\Delta}, q')$ *such that* $f'(S,q'),(q',v) \models \varphi$*. Let* f *be the strategy defined by* $f(q) = a_c$ *and* $f(q \xrightarrow{a_c} \rho) = f'(\rho)$ *for any* ρ *run starting in state* q'*. We have* $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *and* $f(S,q),(q,v) \models [a_c]\, \varphi \wedge \langle \gamma \rangle\, \mathtt{tt}$*.*

– $\Phi \stackrel{\text{def}}{=} \varphi\, [\delta\rangle_s\, \psi$*:*

⇒ *First assume* $\gamma = \lambda$*. This implies that there exists* $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *such that* $f(S,q),(q,v) \models (\varphi\, [\delta\rangle_s\, \psi) \wedge \langle \lambda \rangle\, \mathtt{tt}$*. Hence there is a time step* $f(S,q), s \xrightarrow{t} q^t$ *(with* $t \in \mathbb{R}_+$*) such that* $f(S,q),(q^t, v+t) \models \psi$ *and for all* $t' < t$ $f(S,q),(q^{t'}, v+t') \models \varphi$*. In any of the states* $(q^{t'}, v+t')$ *with* $t' < t$ *the strategy is "delay" and thus they all satisfy* $\langle \lambda \rangle\, \mathtt{tt}$*. By induction hypothesis, this implies that all the states* $S^{q^{t'}}_{\Delta,\tau-t'},(q^{t'}, v+t')$ *satisfy* $\overline{\varphi}^{\lambda}$*.*
Now the strategy in q^t *is either "delay" or do a controllable action; hence* $(q^t, v+t) \models \langle \lambda \rangle\, \mathtt{tt} \vee (\bigvee_{a_c \in \mathsf{Act}_c} \langle a_c \rangle\, \mathtt{tt})$*. As* $(q^t, v+t)$ *satisfies* ψ *and one the* $\langle \gamma \rangle\, \mathtt{tt}$ *for* $\gamma \in \mathsf{Act}_c \cup \{\lambda\}$*,*

by induction hypothesis, we obtain[15]*:* $S^{q^t}_{\Delta,\tau-t}, (q^t, v+t) \models \overline{\psi}^{\,\gamma}$ *for some* $\gamma \in \mathsf{Act}_c \cup \{\lambda\}$*, and thus* $(q^t, v+t) \models \overline{\psi}$*. Finally* $S^q_{\Delta,\tau}, (q,v) \models \overline{\varphi}^{\,\lambda}\, [\delta\rangle_w\, \overline{\psi}$ *which is exactly the definition of* $\overline{\varphi\, [\delta\rangle_s\, \psi}^{\,\lambda}$*.*

Now assume $\gamma = a_c$*. There exists* $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *such that* $f(GS,q), (q,v) \models (\varphi\, [\delta\rangle_s\, \psi) \wedge \langle a_c\rangle\, \mathtt{tt}$*. This means that* $f(q) = a_c$ *and then no delay is allowed by the strategy. This implies that* ψ *must hold in* (q,v)*. Hence* $f(S,q), (q,v) \models \psi \wedge \langle a_c\rangle\, \mathtt{tt}$*. By induction hypothesis we have:* $S^q_{\Delta,\tau}, (q,v) \models \overline{\psi}^{\,a_c}$ *and as* $\overline{\varphi\, [\delta\rangle_s\, \psi}^{\,a_c} = \overline{\psi}^{\,a_c}$ *we are done.*

$\Leftarrow$ *Assume* $S^q_{\Delta,\tau}, (q,v) \models \overline{\varphi\, [\delta\rangle_s\, \psi}^{\,\gamma}$ *and* $\gamma = \lambda$*. By definition of* $\overline{\varphi\, [\delta\rangle_s\, \psi}^{\,\gamma}$ *this means that* $S^q_{\Delta,\tau}, (q,v) \models \overline{\varphi}^{\,\lambda}\, [\delta\rangle_s\, \overline{\psi}$*. This means that there exists some* $t \geq 0$ *such that* $S^q_{\Delta,\tau}, q \xrightarrow{t} q^t$ *and* $S^q_{\Delta,\tau-t}, (q^t, v+t) \models \overline{\psi}$ *and for all* $t' < t$*,* $S^q_{\Delta,\tau-t'}, (q^{t'}, v+t') \models \overline{\varphi}^{\,\lambda}$*. By induction hypothesis this means there exists* $f_t \in \mathsf{Contr}(S^q_{\Delta,\tau-t}, q^t)$ *such that* $f_t(S,q^t), (q^t, v+t) \models \psi \wedge \langle\gamma\rangle\, \mathtt{tt}$ *for* $\gamma \in \mathsf{Act}_c \cup \{\lambda\}$ *and there exists* $f_{t'} \in \mathsf{Contr}(S^q_{\Delta,\tau-t}, q^{t'})$ *such that* $f_{t'}(S,q^{t'}), (q^{t'}, v+t') \models \varphi \wedge \langle\lambda\rangle\, \mathtt{tt}$*. Let* f *be the strategy defined by:* $f(q \xrightarrow{t'} q^{t'}) = \lambda$ *for any* $t' < t$*,* $f(q \xrightarrow{t} q^t) = \gamma$ *if* $f(q \xrightarrow{t} q^t) = \gamma$*. Then* $f(S,q), (q,v) \models \varphi\, [\delta\rangle_s\, \psi \wedge \langle\lambda\rangle\, \mathtt{tt}$*.*

Assume $\gamma \in \mathsf{Act}_c$*. If* $S^q_{\Delta,\tau}, (q,v) \models \overline{\varphi\, [\delta\rangle_s\, \psi}^{\,\gamma}$*. By definition of* $\overline{\varphi\, [\delta\rangle_s\, \psi}^{\,\gamma}$ *this means that* $S^q_{\Delta,\tau}, (q,v) \models \overline{\psi}^{\,\gamma}$ *and by induction hypothesis this implies there exists a strategy* f *such that* $f(S,q), (q,v) \models \psi \wedge \langle\gamma\rangle\, \mathtt{tt}$*. Thus* $f(S,q), (q,v) \models \varphi\, [\delta\rangle_s\, \psi \wedge \langle\gamma\rangle\, \mathtt{tt}$

– $\Phi \stackrel{\text{def}}{=} [\delta]\, \varphi$*:*

$\Rightarrow$ *First assume* $\gamma \in \mathsf{Act}_c$ *and there exists* $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *such that* $f(S,q), (q,v) \models [\delta]\, \varphi \wedge \langle\gamma\rangle\, \mathtt{tt}$*. This implies* $f(S,q), (q,v) \models \varphi \wedge \langle\gamma\rangle\, \mathtt{tt}$ *and by induction hypothesis we have:* $S^q_{\Delta,\tau}, (q,v) \models \overline{\varphi}^{\,\gamma}$ *which is by equation (5)* $\overline{[\delta]\, \varphi}^{\,\gamma}$*.*
Assume $\gamma = \lambda$ *and there exists* $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *such that* $f(S,q), (q,v) \models [\delta]\, \varphi$*,* i.e.*, for any transition* $f(S,q), q \xrightarrow{t} q^t$ *(with* $t \in \mathbb{R}_+$*), we have* $f(S,q), (q^t, v+t) \models \varphi$*. We distinguish two cases:*

[15] In the sequel we assume $\tau - t$ stands for $\max(0, \tau - t)$.

- *either the strategy is to delay for ever in* $S^q_{\Delta,\tau}$ *from* q. *In this case we have* $f(S,q),(q^t,v+t) \models [\delta]\,\varphi \wedge \langle\lambda\rangle\,\mathtt{tt}$ *for every* $t \geq 0$. *This implies by induction hypothesis* $S^q_{\Delta,\tau-t}(q^t,v+t) \models \overline{\varphi}^{\,\lambda}$ *for every* $t \geq 0$ *and thus* $S^q_{\Delta,\tau},(q,v) \models \overline{\varphi}^{\,\lambda}\,[\delta\rangle_w\,\psi$ *for any* ψ *and hence* $S^q_{\Delta,\tau},(q,v) \models [\delta]\,\overline{\varphi}^{\,\lambda}$. *This in turn implies* $\overline{\varphi}^{\,\lambda}\,[\delta\rangle_w\,\psi'$ *for any* ψ'.
- *or there is some* $t \geq 0$ *such that* $f(q \xrightarrow{\ t\ } q^t) = a_c$. *We then have* $f(S,q),(q^{t'},v+t') \models \varphi \wedge \langle\lambda\rangle\,\mathtt{tt}$ *for any* $t' < t$ *and* $f(S,q),(q^t,v+t) \models \varphi \wedge \langle a_c\rangle\,\mathtt{tt}$. *By induction hypothesis we obtain* $S^{q^{t'}}_{\Delta,\tau-t'},(q^{t'},v+t') \models \overline{\varphi}^{\,\lambda}$ *for any* $t' < t$ *and* $S^q_{\Delta,0},(q^t,v+t) \models \overline{\varphi}^{\,a_c}$. *This implies* $S^q_{\Delta,\tau},(q,v) \models \overline{\varphi}^{\,\lambda}[\delta\rangle_w\overline{\varphi}^{\,a_c}$ *which is by equation (5)* $\overline{[\delta]\,\varphi}^{\,\lambda}$.

$\Leftarrow$ *Assume* $\gamma \in \mathsf{Act}_c$. *Then* $S^q_{\Delta,\tau},(q,v) \models \overline{\varphi}^{\,\gamma}$ *entails that there exists some* $f \in \mathsf{Contr}(S^q_{\Delta,\tau},q)$ *such that* $f(S,q),(q,v) \models \varphi \wedge \langle\gamma\rangle\,\mathtt{tt}$. *This means that* $f(q) = \gamma$ *and that no delay is indeed allowed by the strategy* f. *Then we clearly have* $f(S,q),(q,v) \models [\delta]\,\varphi \wedge \langle\gamma\rangle\,\mathtt{tt}$.
Assume $\gamma = \lambda$ *and* $S^q_{\Delta,\tau},(q,v) \models \overline{\varphi}^{\,\lambda}\,[\delta\rangle_w\,\overline{\varphi}^{\,a_c}$ *for some* $a_c \in \mathsf{Act}_c$. *We distinguish two cases:*

- $S^q_{\Delta,\tau},(q,v) \models [\delta]\,\overline{\varphi}^{\,\lambda}$. *Then for any* $t \in \mathbb{R}_+$, *we have that* $S^q_{\Delta,\tau} : q \xrightarrow{\ t\ } q^t$ *implies* $S^q_{\Delta,\tau},(q^t,v+t) \models \overline{\varphi}^{\,\lambda}$ *and then by induction hypothesis there exists* $f_t \in \mathsf{Contr}(S^q_{\Delta,\tau-t},q^t)$ *such that* $f_t(S,q^t),(q^t,v+t) \models \varphi$ *and* $f_t(q^t) = \lambda$. *Let* f *be the strategy defined by* $f(q \xrightarrow{\ t\ } q^t) = \lambda$ *and* $f(q \xrightarrow{\ t\ } \rho) = f_t(\rho)$ *for any run* ρ *starting in state* q^t. *Clearly* f *belongs to* $\mathsf{Contr}(S^q_{\Delta,\tau},q)$. *And we have* $f(S,q),(q,v) \models [\delta]\,\varphi$.
- *There exists* $t \in \mathbb{R}_+$ *such that* $S^q_{\Delta,\tau} : q \xrightarrow{\ t\ } q^t$ *and* $S^q_{\Delta,\tau-t},(q^t,v+t) \models \overline{\varphi}^{\,a_c}$ *for some* $a_c \in \mathsf{Act}_c$. *By induction hypothesis there exists* $f_t \in \mathsf{Contr}(S^q_{\Delta,\tau-t},q^t)$ *such that* $f_t(S,q^t),(q^t,v+t) \models \varphi \wedge \langle a_c\rangle\,\mathtt{tt}$. *Clearly* $f_t(q^t) = a_c$ *and* f_t *forbids time elapsing from* q^t. *Moreover the induction hypothesis applied to states* $q^{t'}$ *with* $t' < t$ *gives that there exists* $f_{t'} \in \mathsf{Contr}(S^q_{\Delta,\tau-t},q^{t'})$ *such that* $f_{t'}(S,q^{t'}),(q^{t'},v+t') \models \varphi$ *and* $f_{t'}(q^{t'}) = \lambda$. *Let* f *be the strategy defined by:* $f(q \xrightarrow{\ t'\ } q^{t'}) = \lambda$ *for any* $t' < t$, $f(q \xrightarrow{\ t\ } q^t) = a_c$ *and* $f(q \xrightarrow{\ t''\ } \rho) = f_{t''}(\rho)$ *for* $t'' \leq t$. *This strategy allows us to deduce* $f(S,q),(q,v) \models [\delta]\,\varphi$.

– $\Phi \stackrel{\text{def}}{=} \bigvee_i \varphi_i$: *Direct.*

– $\Phi \stackrel{\text{def}}{=} \bigwedge_i \varphi_i$:

⇒ *Assume there exists* $f \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *such that* $f(S,q),(q,v) \models \bigwedge_i \alpha_i \wedge \langle\gamma\rangle\, \mathtt{tt}$. *Then we have* $f(S,q),(q,v) \models \alpha_i \wedge \langle\gamma\rangle\, \mathtt{tt}$ *for any* i. *By h.i. we have* $S^q_{\Delta,\tau},(q,v) \models \overline{\alpha_i}^{\,\gamma}$ *for any* i, *and then* $S^q_{\Delta,\tau},(q,v) \models \bigwedge_i \overline{\alpha_i}^{\,\gamma}$.

⇐ *Assume* $S^q_{\Delta,\tau},(q,v) \models \bigwedge_i \overline{\varphi}^{\,\gamma}$. *Then by induction hypothesis we have that there exists some* $f_i \in \mathsf{Contr}(S^q_{\Delta,\tau}, q)$ *such that* $f_i(S,q),(q,v) \models \alpha_i \wedge \langle\gamma\rangle\, \mathtt{tt}$. *It remains to construct a strategy* f *by collecting the strategies* f_i*'s. This is possible because* φ *belongs to* L_c^{det}, *indeed any term* α_i *is prefixed by a modality with a different label of* $\mathsf{Act} \cup \{\delta\}$ *and then the union of the strategies* f_i*'s provides a strategy* f *that belongs to* $\mathsf{Contr}(S^q_{\Delta,\tau}, q)$. *This gives the result.*

Then this entails that the Lemma holds for any L_c^{det} *formula without fixpoint. But this clearly entails that it also holds for full* L_c^{det}. *Indeed consider two states* q *and* q' *which satisfy the same formulas without fixpoint. If* q *does not belong to the greatest fixpoint of a an equation* $Z \stackrel{\text{def}}{=} \Psi_Z$, *then it entails that this state does not satisfy some unfolding of the formula* Ψ_Z *(with where the first occurrences of* Z *have been replaced by* $\mathtt{tt}$*), then this formula does not hold for the state* q'. □