From Timed Automata to Logic — and Back *

François Laroussinie[1], Kim G. Larsen[1], Carsten Weise[2]

[1] **BRICS*****, Aalborg Univ., Denmark
[2] Aachen Univ., Germany

Abstract. In this paper, we define a timed logic L_ν which is sufficiently expressive that we for any timed automaton may construct a single *characteristic* L_ν formula uniquely characterizing the automaton up to timed bisimilarity.
Also, we prove decidability of the *satisfiability* problem for L_ν with respect to given bounds on the number of clocks and constants of the timed automata to be constructed.

1 Introduction

One of the most successful techniques for *automatic verification* is that of *model–checking*; i.e. a property is given as a formula of a propositional temporal logic and automatically compared with an automaton representing the actual behaviour of the system. Extremely efficient model–checking algorithms have been obtained for *finite* automata with respect to the branching–time temporal logics CTL [7, 22, 8] and the modal μ–calculus [17, 4, 10, 9, 3, 24]. In the last few years, model–checking has been extended to real–time systems, with time considered to be a dense linear order. A timed extension of finite automata through addition of a finite set of real–valued clocks has been put forward [2], and the corresponding model–checking problem has been proven decidable for a number of timed logics including timed extensions of CTL (TCTL) [1] and a timed μ–calculus (T_μ) [13].

In this paper we continue this transfer of existing techniques from the setting of finite (untimed) automata to that of timed automata. In particular a timed logic L_ν is put forward, which is sufficiently expressive that we for any timed automaton may (effectively) construct a single *characteristic* L_ν formula uniquely characterizing the automaton up to timed bisimilarity. The construction is a timed extension of those in [5, 12, 16], and reduces timed bisimilarity between automata to a model–checking problem, which — when combined with the model–checking algorithm for L_ν — yields an alternative algorithm for timed bisimulation compared with [6]. In addition, characteristic formula constructions may be given for other behavioural preorders [19, 11], immediately yielding decision procedures for these relationships as well. Secondly, we prove decidability of

* This work has been supported by the European Communities under CONCUR2, BRA 7166

*** **B**asic **R**esearch in **C**omputer **S**cience, Centre of the Danish National Research Foundation.

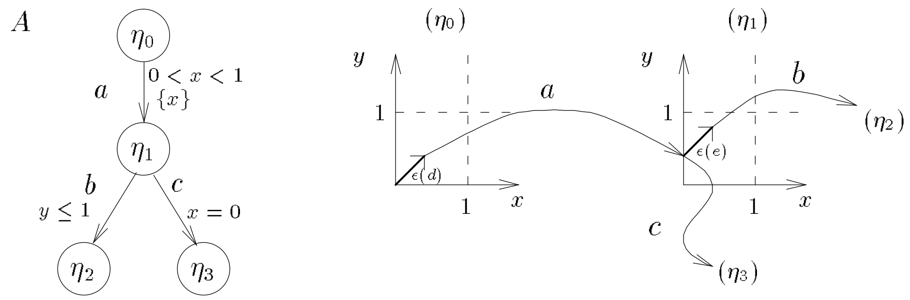


Fig. 1. An automaton and its behaviour

bounded satisfiable for L_ν. That is, we present a model–construction algorithm, which given a formula of L_ν and bounds k and M will synthesize a timed automaton with no more than k clocks and no clock being compared with constants greater than M (provided any such exits).

Combining the characteristic formula construction with the bounded model–construction algorithm enables us to decide whether an automaton can be simplified in terms of number of clocks and constants used for comparison.

A full version can be found in [18].

2 Timed Automata

Let $\mathcal{A}$ be a fixed set of actions ranged over by $a, b, c, \ldots$. We denote by $\mathbf{N}$ the set of natural numbers and by $\mathbf{R}$ the set of non–negative real numbers. $\mathcal{D}$ denotes the set of delay actions $\{\epsilon(d) \mid d \in \mathbf{R}\}$, and $\mathcal{L}$ denotes the union $\mathcal{A} \cup \mathcal{D}$. If C is a set of clocks, $\mathcal{B}(C)$ denotes the set of formulas built using boolean connectives over atomic formulas of the form $x \leq m$, $m \leq x$, $x \leq y + m$ and $y + m \leq x$ with $x, y \in C$ and $m \in \mathbf{N}$. Moreover $\mathcal{B}_M(C)$ denotes the subset of $\mathcal{B}(C)$ with no constant greater than M.

Definition 1. A timed automaton A is a tuple $\langle \mathcal{A}, N, \eta_0, C, E \rangle$ where $\mathcal{A}$ is a finite set of actions, N is a finite set of nodes, η_0 is the initial node, C is a finite set of clocks, and $E \subseteq N \times N \times \mathcal{A} \times 2^C \times \mathcal{B}(C)$ corresponds to the set of edges. $e = \langle \eta, \eta', a, r, b \rangle \in E$ represents an edge from the node η to the node η' with action a, r denoting the set of clocks to be reset and b is the enabling condition over the clocks of A.

Informally, the system starts at node η_0 with all its clocks initialized to 0. The values of the clocks increase synchronously with time. At any time, the automaton whose current node is η can change node by following an edge $\langle \eta, \eta', a, r, b \rangle \in E$ provided the current values of the clocks satisfy b. With this transition the clocks in r get reset to 0.

A *time assignment* v for C is a function from C to $\mathbf{R}$. We denote by $\mathbf{R}^C$ the set of time assignments for C. For $v \in \mathbf{R}^C$, $x \in C$ and $d \in \mathbf{R}$, $v + d$ denotes the time assignment which maps each clock x in C to the value $v(x)+d$. For $C' \subseteq C$, $[C' \mapsto 0]v$ denotes the assignment for C which maps each clock in C' to the value 0 and agrees with v over $C \backslash C'$. Given a condition $b \in \mathcal{B}(C)$ and a time assignment $v \in \mathbf{R}^C$, $b(v)$ is a boolean value describing whether b is satisfied by v or not. Finally a k–clock automaton is a timed automaton $\langle \mathcal{A}, S, \eta_0, C, E \rangle$ such that $|C| = k$.

A *state* of an automaton A is a pair $\langle \eta, v \rangle_A$ where η is a node of A and v a time assignment for C. The initial state of A is $\langle \eta_0, v_0 \rangle_A$ where v_0 is the time assignment mapping all clocks in C to 0. The semantics of A is given by a labelled transition system $\mathcal{M}_A = \langle \Sigma_A, \mathcal{L}, \sigma_0, \longrightarrow_A \rangle$, where Σ_A is the set of states of A, σ_0 is the initial state $\langle \eta_0, v_0 \rangle_A$, and $\longrightarrow_A$ is the transition relation defined as follows: $\langle \eta, v \rangle \xrightarrow{a}_A \langle \eta', v' \rangle$ iff $\exists r, b.$ s.t. $\langle \eta, \eta', a, r, b \rangle \in E \wedge b(v) \wedge v' = [r \rightarrow 0]v$ and $\langle \eta, v \rangle \xrightarrow{\epsilon(d)}_A \langle \eta', v' \rangle$ iff $\eta = \eta'$ and $v' = v + d$

We may now apply the standard notion of bisimulation [20, 21] to the labelled transition systems determined by two automata A and B. Letting s_A and s_B range over states of respectively A and B, *strong timed bisimulation* $\sim$ is defined as the largest symmetric relation over $\Sigma_A \times \Sigma_B$ such that whenever $s_A \sim s_B$, $\ell \in \mathcal{A} \cup \mathcal{D}$ and $s_A \xrightarrow{\ell}_A s'_A$ then there exists s'_B such that $s_B \xrightarrow{\ell}_B s'_B$ and $s'_A \sim s'_B$. We say that A and B are *strong timed bisimular* if their initial states are strong bisimilar.

Example 1. Consider the automaton A of Figure 1. The two coordinate systems in the right part of the Figure indicate (some of) the states of A. Each point of the coordinate systems represents a unique time assignment, with the left (resp. right) coordinate system representing states involving the node η_0 (resp. η_1). In the Figure we have indicated some transition sequences (with $d < 1$ and $e + d \leq 1$).

3 Timed Modal Logic L_ν

We consider a dense–time logic L_ν with clocks and recursion. This logic may be seen as a certain fragment of the μ–calculus T_μ presented in [13].

Definition 2. Let K a finite set of clocks, Id a set of identifiers and k an integer. The set L_ν of formulae over K, Id and k is generated by the abstract syntax with φ and ψ ranging over L_ν:

$$\varphi ::= \mathtt{tt} \mid \mathtt{ff} \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \exists\!\!\exists\, \varphi \mid \forall\!\!\forall\, \varphi \mid \langle a \rangle\, \varphi \mid [a]\, \varphi$$
$$\mid x \;\mathsf{in}\; \varphi \mid x + n \bowtie y + m \mid x \bowtie m \mid Z$$

where $a \in \mathcal{A}$; $x, y \in K$; $n, m \in \{0, 1, \ldots, k\}$; $\bowtie \in \{=, <, \leq, >, \geq\}$ and $Z \in \mathsf{Id}$.

The meaning of the identifiers is specified by a declaration $\mathcal{D}$ assigning a formula of L_ν to each identifier. When $\mathcal{D}$ is understood we write $Z \stackrel{\mathrm{def}}{=} \varphi$ for $\mathcal{D}(Z) = \varphi$. The K clocks are called *formula clocks* and a formula φ is said to be *closed* if every formula clock x occurring in φ is in the scope of an "x in …" operator.

Given a timed automaton $A = \langle \mathcal{A}, N, \eta_0, C, E \rangle$, we interpret the L_ν formulas over an *extended state* $\langle \eta, vu \rangle_{A^+}$ where $\langle \eta, v \rangle_A$ is a state of A and u a time assignment for K. Transitions between extended states are defined by: $\langle \eta, vu \rangle_{A^+} \xrightarrow{\epsilon(d)} \langle \eta', v+d\; u+d \rangle_{A^+}$ and $\langle \eta, vu \rangle_{A^+} \xrightarrow{a} \langle \eta', v'u' \rangle_{A^+}$ iff $\langle \eta, v \rangle_A \xrightarrow{a} \langle \eta', v' \rangle_A$ and $u = u'$. Formally, the satisfaction relation between extended states and formulas is defined as follows:

Definition 3. Let A be a timed automaton and $\mathcal{D}$ a declaration. The satisfaction relation $\models_{\mathcal{D}}$ is the largest relation satisfying the following implications :

$$
\begin{array}{rcl}
\langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} \mathsf{tt} & \Rightarrow & \text{true} \\
\langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} \mathsf{ff} & \Rightarrow & \text{false} \\
\langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} \varphi \wedge \psi & \Rightarrow & \langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} \varphi \text{ and } \langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} \psi \\
\langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} \exists\, \varphi & \Rightarrow & \exists d \in \mathbf{R}.\ \ \langle \eta, v{+}d\;\; u{+}d \rangle_{A^+} \models_{\mathcal{D}} \varphi \\
\langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} \langle a \rangle\, \varphi & \Rightarrow & \exists\, \langle \eta', v' \rangle_A.\ \ \langle \eta, v \rangle_A \xrightarrow{a} \langle \eta', v' \rangle_A \text{ and } \langle \eta', v'\, u \rangle_{A^+} \models_{\mathcal{D}} \varphi \\
\langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} x{+}m{\bowtie}y{+}n & \Rightarrow & u(x) + m \bowtie u(y) + n \\
\langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} x \;\mathsf{in}\; \varphi & \Rightarrow & \langle \eta, v\, u' \rangle_{A^+} \models_{\mathcal{D}} \varphi \text{ where } u' = [\{x\} \to 0]u \\
\langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} Z & \Rightarrow & \langle \eta, v\, u \rangle_{A^+} \models_{\mathcal{D}} \mathcal{D}(Z)
\end{array}
$$

Any relation satisfying the above implications is called a *satisfiability* relation. It follows from standard fixpoint theory [23] that $\models_{\mathcal{D}}$ is the union of all satisfiability relations and that the above implications in fact are biimplications for $\models_{\mathcal{D}}$. We say that A satisfies a closed L_ν formula φ and write $A \models \varphi$ when $\langle \eta_0, v_0\, u \rangle_{A^+} \models_{\mathcal{D}} \varphi$ for any u. Note that if φ is closed, then $\langle \eta, vu \rangle_{A^+} \models_{\mathcal{D}} \varphi$ iff $\langle \eta, vu' \rangle_{A^+} \models_{\mathcal{D}} \varphi$ for any $u, u' \in \mathbf{R}^K$.

The real–time interval modality $\exists]m;n[$ (resp. $\forall]m;n[$) introduced in [14] which denotes existential (resp. universal) quantification over delay between m and n, can be defined in L_ν [4]. A formula is called a *q-clocks formula* if it contains no more than q formula clocks.

Example 2. The initial state $\langle \eta_0, v_0\, u_0 \rangle$ of the automaton from Figure 1 satisfies the following L_ν formula φ:

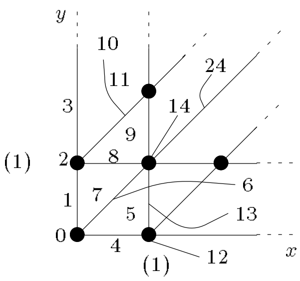
$$\varphi = \exists\,]0;1[\,\langle a \rangle\,\Big[\big(\langle c \rangle\,\mathsf{tt}\big) \wedge \big(\forall\,]0;1[\,[c]\,\mathsf{ff}\big) \wedge \big(\exists\,]0;1[\,\langle b \rangle\,\mathsf{tt}\big) \wedge \big(\exists\,]0;1[\,[b]\,\mathsf{ff}\big)\Big] \qquad (1)$$

4 Model Checking

The model-checking problem for L_ν consists in deciding if a given timed automaton A satisfies a given specification φ in L_ν. This problem is decidable using the region technique of Alur and Dill [2, 1] which provides an abstract semantics of timed automata in the form of finite labelled transition systems with the truth value of L_ν formulas being maintained. The basic idea is that, given a timed automaton A, two states $\langle \eta, v_1 \rangle_A$ and $\langle \eta, v_2 \rangle_A$ which are close enough with respect to their clocks values (we will say that v_1 and v_2 are in the same *region*) can perform the same actions, and two extended states $\langle \eta, v_1\, u_1 \rangle_{A^+}$ and $\langle \eta, v_2\, u_2 \rangle_{A^+}$

---

[4] $\exists]m;n[\varphi \stackrel{\text{def}}{=} x\,\mathsf{in}\big(\exists(x > m \wedge x < n \wedge \varphi)\big)$ and $\forall]m;n[\varphi \stackrel{\text{def}}{=} x\,\mathsf{in}\big(\forall(x \leq m \vee x \geq n \vee \varphi)\big)$



$succ(\gamma_0) = \gamma_6$
$succ(\gamma_5) = \gamma_{13}$
$\ldots$

$\beta(\gamma_0) = (x = 0 \wedge y = 0)$
$\beta(\gamma_8) = (0 < x < 1 \wedge y = 1)$

$\beta(\gamma_6) = \begin{array}{l} 0 < x < 1 \wedge \\ 0 < y < 1 \wedge \\ y = x \end{array}$

$\beta(\gamma_7) = \begin{array}{l} 0 < x < 1 \wedge \\ 0 < y < 1 \wedge \\ y > x \end{array}$

Fig. 2. $\mathcal{R}_k^C$ with $C = \{x, y\}$ and $k = 1$

where $v_1\ u_1$ and $v_2\ u_2$ are in the same region, satisfy the same L_ν formulas. In fact the regions are defined as equivalence classes of a relation $\doteq$ over time assignments [13]. Formally, given C a set of clocks and k an integer, we say $u \doteq v$ if and only if u and v satisfy the same conditions of $\mathcal{B}_k(C)$. $[u]$ denotes the region which contains the time assignment u. $\mathcal{R}_k^C$ denotes the set of all regions for a set C of clocks and the maximal constant k. From a decision point of view it is important to note that $\mathcal{R}_k^C$ is finite.

For a region $\gamma \in \mathcal{R}_k^C$, we can define $b(\gamma)$ as the truth value of $b(u)$ for any u in γ. Conversely given a region γ, we can easily build a formula of $\mathcal{B}(C)$, called $\beta(\gamma)$, such that $\beta(\gamma)(u) = \mathtt{tt}$ iff $u \in \gamma$. Thus, given a region γ', $\beta(\gamma)(\gamma')$ is mapped to the value $\mathtt{tt}$ precisely when $\gamma = \gamma'$. Finally, note that $\beta(\gamma)$ itself can be viewed as a L_ν formula.

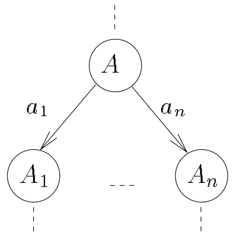
Given a region $[u]$ in $\mathcal{R}_k^C$ and $C' \subseteq C$ we define the following reset operator: $[C' \to 0][u] = [[C' \to 0]u]$. Moreover given a region γ, we can define the successor region of γ (denoted by $succ(\gamma)$): Informally the change from γ to $succ(\gamma)$ correspond to the minimal elapse of time which can modify the enabled actions of the current state(a formal definition is given in [18]).

We denote by γ^l the l^{th} successor region of γ (i.e. $\gamma^l = succ^l(\gamma)$). From each region γ, it is possible to reach a region γ' s.t. $succ(\gamma') = \gamma'$, and we denote by l_γ the required number of step s.t. $\gamma^{l_\gamma} = succ(\gamma^{l_\gamma})$.

Example 3. The Figure 2 gives an overview of the set of regions defined by two clocks x and y, and the maximal constant 1. In this case there are 32 different regions. In general successor regions are determined by following 45^o lines upwards to the right.

Given a timed automata $A = \langle \mathcal{A}, N, \eta_0, C, E\rangle$, let k_A be the maximal constant occurring in the enabling condition of the edges E. Then for any $k \geq k_A$ we can define a symbolic semantics of A over symbolic states $[\eta, \gamma]_A$ where $\eta \in N$ and $\gamma \in \mathcal{R}_k^C$ as follows: for any $[\eta, \gamma]$ we have $[\eta, \gamma]_A \xrightarrow{a} [\eta', \gamma']_A$ iff $\exists\ u \in \gamma,\ \langle \eta, u\rangle_A \xrightarrow{a} \langle \eta', u'\rangle_A$ and $u' \in \gamma'$.

Consider now L_ν with respect to formula clock set K and maximal constant k_L. Also consider a given timed automaton $A = \langle \mathcal{A}, N, \eta_0, C, E\rangle$ (s.t. K and C are disjoint). Then an *extended symbolic state* is a pair $[\eta, \gamma]_{A^+}$ where $\eta \in N$ and $\gamma \in \mathcal{R}_k^{C^+}$ with $C^+ = C \cup K$ and $k = max(k_A, k_L)$. We can define the



$$\Phi(A) = \bigwedge_{i=1}^{n} \langle a_i \rangle\, \Phi(A_i) \;\wedge\; \bigwedge_{a} [a] \Big(\bigvee_{i.a_i=a} \Phi(A_i) \Big)$$

Fig. 3. Characteristic formula for finite automata.

symbolic semantics for L_ν, i.e. the truth value of L_ν formulas over the extended symbolic state. Due to space limitation we only give the two main implications defining the *symbolic satisfiability relation* $\vdash_{\mathcal{D}}$ [5]:

$$\begin{array}{l} [\eta, \gamma]_{A^+} \vdash_{\mathcal{D}} \exists\, \varphi \;\;\Rightarrow\;\; \exists l \in \mathbf{N}.\;\; [\eta, succ^l(\gamma)]_{A^+} \vdash_{\mathcal{D}} \varphi \\ [\eta, \gamma]_{A^+} \vdash_{\mathcal{D}} \langle a \rangle\, \varphi \;\;\Rightarrow\;\; \exists\, [\eta, \gamma_{|C}]_A \xrightarrow{a} [\eta', \gamma'_{|C}]_A \;\mathbf{s.t.}\;\; \gamma'_{|K} = \gamma_{|K} \;\wedge\; [\eta', \gamma']_{A^+} \vdash_{\mathcal{D}} \varphi \end{array}$$

We have the following important result: Let φ be a formula of L_ν, and let $\langle \eta, v\ u \rangle_{A^+}$ be an extended state over some timed automaton A, then we have [6]:

$$\langle \eta, v\ u \rangle_{A^+} \models_{\mathcal{D}} \varphi \quad \text{if and only if} \quad [\eta, [v \cdot u]]_{A^+} \vdash_{\mathcal{D}} \varphi$$

It follows that the model checking problem for L_ν is decidable since, given $\varphi \in L_\nu$, it suffices to check the truth value of any given L_ν formula φ with respect to a finite transition system corresponding to the extended symbolic semantics of A.

5 Characteristic Properties

First let us recall the characteristic formula construction for finite automata [16, 12, 5] (see Figure 3). The construction defines the characteristic formula $\Phi(A)$ of a node A in terms of similar characteristic formulas of the derivates $A_1 \ldots A_n$ of A: whenever A has an a_i–transition to A_i this is reflected in $\Phi(A)$ by addition of a conjunct $\langle a_i \rangle \Phi(A_i)$. To characterize A up to strong bisimilarity $\Phi(A)$ contains in addition a conjunct $[a]\Psi_a$ for each action a, where Ψ_a is a disjunction over all a–transitions out of A. In general the definitions of characteristic formulas $\Phi(A)$ constitutes a simultaneous recursive definition (as the automaton may have cycles), and to obtain the desired characterization the solution sought is the maximum one. For timed automata the characteristic formula construction must necessarily take account of the time assignment in addition to the actual node. Thus, for a timed automaton $A = \langle \mathcal{A}, N, \eta_0, C, E \rangle$, we shall define characteristic formulas of the form $\Phi(\eta, \gamma)$, where η is a node of A

[5] $\gamma_{|C}$ (resp. $\gamma_{|K}$) denotes the set of time-assignments in γ restricted to the automata (resp. formula) clocks.

[6] where $v \cdot u$ is the time assignment over $C \cup K$ such that $(v \cdot u)(x) = v(x)$ if $x \in C$ and $(v \cdot u)(x) = u(x)$ if $x \in K$.

and γ is a region over the clocks of A. The construction of $\Phi(\eta,\gamma)$ follows closely the pattern from the finite automa case. However, we first need to be able to determine the (a–) edges out of η which are enabled in the region γ. Given an edge $e = \langle \eta, \eta', a, r, b\rangle$ in E, η_e (resp. η'_e, a_e, r_e, b_e) denotes η (resp. η', a, r, b). Given $\eta \in N$ and $\gamma \in \mathcal{R}^C_{k_A}$, we define $E(\eta,\gamma) = \{e \mid \eta_e = \eta \text{ and } b_e(\gamma) = \mathtt{tt}\}$ and $E(\eta,\gamma,a) = \{e \in E(\eta,\gamma) \mid a_e = a\}$. Thus, $E(\eta,\gamma)$ (resp. $E(\eta,\gamma,a)$) is the set of all enabled transitions (resp. a-transitions) from $[\eta,\gamma]_A$. We may now present the characteristic formula construction for timed automata:

Definition 4. Let A be a timed automaton $\langle \mathcal{A}, N, \eta_0, C, E\rangle$. For any region γ in $\mathcal{R}^C_{k_A}$, and node η in N, we introduce an identifier $\Phi(\eta,\gamma)$ (the characteristic formula) associated with the symbolic state $[\eta,\gamma]_A$. The definition (declaration) for $\Phi(\eta,\gamma)$ is:

$$\Phi(\eta,\gamma) \stackrel{\text{def}}{=} \left(\begin{array}{c} \bigwedge_{e\in E(\eta,\gamma)} \langle a_e\rangle \Big(r_e \,\mathsf{in}\, \Phi(\eta'_e, r_e(\gamma))\Big) \wedge \bigwedge_a [a] \Big(\bigvee_{e\in E(\eta,\gamma,a)} \Big(r_e \,\mathsf{in}\, \Phi(\eta'_e, r_e(\gamma))\Big)\Big) \\ \wedge \,\forall\!\!\!\forall \Big(\bigwedge_{l=0..l_\gamma} \beta(\gamma^l) \Rightarrow \Phi(\eta,\gamma^l)\Big) \end{array} \right)$$

We denote by Id_A the set of identifiers $\Phi(\eta,\gamma)$ and by $\mathcal{D}_A$ the corresponding declaration.

Note that the declaration for $\Phi(\eta,\gamma)$ is not quite a L_ν formula due to the presence of implication. However, it is easy to transform it into an equivalent L_ν formula because the negation of $\beta(\gamma)$ can be expressed in L_ν. Moreover ($r \,\mathsf{in}\, \varphi$) is an abbreviation for $(c_1 \,\mathsf{in}\, (c_2 \,\mathsf{in}\, \ldots (c_n \,\mathsf{in}\, \varphi)))$ whenever r is $\{c_1,\ldots,c_n\}$. Finally $r(\gamma)$ denotes $[r \rightarrow 0]\gamma$. Note that $\mathcal{D}_A$ uses no more than $|C|$ formula clocks.

The declaration for $\Phi(\eta,\gamma)$ contains three groups of conjunctions the two first of which are closely related to the characteristic formula construction for finite automata. The first group contains a $\langle a_e\rangle$–formula for any edge e, which is enabled at η in the region γ. Following this edge clearly takes the automaton to the extended state $[\eta'_e, r_e(\gamma)]$. The second group of conjuncts contains for each action a a formula of the type $[a]\Psi_a$, where Ψ is a disjunction over all a–labelled edges being enabled at η in the region γ. Whereas the two first groups exhaustively characterizes the action behaviour of the extended state $[\eta,\gamma]$, the third conjunct is a $\forall\!\!\!\forall$–formula dealing with all delay transitions by requiring that any delay leading to a particular successor region γ^l should satisfy the corresponding characteristic formula.

Example 4. Reconsider the timed automaton A described in Example 1 and the corresponding regions from Example 3. Below we give the declaration of some of the characteristic formulas. We define $\varphi_{nil} \stackrel{\text{def}}{=} \bigwedge_a [a]\,\mathtt{ff}$ and we denote $\beta(\gamma_i)$ by β_i. We have:

$$\begin{aligned} \Phi(\eta_0,\gamma_0) \stackrel{\text{def}}{=}\; & \varphi_{nil} \wedge \forall\!\!\!\forall \Big[(\beta_0 \Rightarrow \Phi(\eta_0,\gamma_0)) \wedge (\beta_6 \Rightarrow \Phi(\eta_0,\gamma_6)) \wedge (\beta_{14} \Rightarrow \Phi(\eta_0,\gamma_{14})) \\ & \wedge (\beta_{24} \Rightarrow \Phi(\eta_0,\gamma_{24}))\Big] \\ \Phi(\eta_0,\gamma_6) \stackrel{\text{def}}{=}\; & \langle a\rangle\, x \,\mathsf{in}\, \Phi(\eta_1,\gamma_1) \wedge [a]\, x \,\mathsf{in}\, \Phi(\eta_1,\gamma_1) \wedge [b]\,\mathtt{ff} \wedge [c]\,\mathtt{ff} \\ & \wedge \forall\!\!\!\forall \Big[(\beta_6 \Rightarrow \Phi(\eta_0,\gamma_6)) \wedge (\beta_{14} \Rightarrow \Phi(\eta_0,\gamma_{14})) \wedge (\beta_{24} \Rightarrow \Phi(\eta_0,\gamma_{24}))\Big] \end{aligned}$$

We have the following Main Theorem the proof of which is given in [18].

Theorem 5. *Let* $A = \langle \mathcal{A}, N, \eta_0, C, E\rangle$ *and* $B = \langle \mathcal{A}, M, \rho_0, K, F\rangle$ *be two timed automata. Then for any* $\rho \in M$, $\eta \in N$, $v \in \boldsymbol{R}^K$ *and* $u \in \boldsymbol{R}^C$: $\langle \rho, v\rangle_B \sim \langle \eta, u\rangle_A$ *iff* $\langle \rho, v\ u\rangle_{B^+} \models_{\mathcal{D}_A} \Phi(\eta, [u])$ *where* $\mathcal{D}_A$ *corresponds to the previous definition of* $\Phi(\eta, \gamma)$ *for each* $\eta \in N$ *and* $\gamma \in \mathcal{R}^C_{k_A}$.

As model–checking of L_ν is decidable we may use the above characteristic formula construction to decide timed bisimilarity between timed automata: to decide if two timed automata are timed bisimilar simply compare one automaton to the characteristic formula of the other.

6 Model Construction

In this section we address the *satisfiability* problem for L_ν. That is we want to decide whether there exists a timed automaton A satisfying a given L_ν–formula φ. The hardness of this problem is illustrated by the following 1-clock formula:

$$\Psi_l \stackrel{\text{def}}{=} \Big(\underbrace{\exists\,]0;\infty[\,\langle a\rangle\ \cdots \exists\,]0;\infty[\,\langle a\rangle}_{l}\Big)\Big[\bigwedge_{i=1..l} \exists\,]0;1[\,\Big(\langle a_i\rangle\,\mathtt{tt} \wedge \bigwedge_{j\neq i}[a_j]\,\mathtt{ff}\Big)\Big]$$

where $l \in \mathbf{N}$. Indeed Ψ_l is satisfiable by some p-clock automata if and only if $l \leq 2p+1$. As a consequence of this remark (see [18]) we cannot deduce the number of clocks in the automaton from the number of clocks in φ. In fact, similar to the results for TCTL and T_μ, we conjecture that the satisfiability problem for L_ν is undecidable.

Instead, we address the following more restricted *bounded satisfiability* problem in which bounds have been placed on both the number of automaton clocks as well as the size of the constants these clocks are compared to: given a formula φ (over a declaration $\mathcal{D}$), a set of clocks C and an integer M, we want to decide (and synthesize) whether there exists a (C, M)–automaton s.t. $A \models_{\mathcal{D}} \varphi$. We have the following main result:

Theorem 6. *The bounded satisfiability problem for* L_ν *is decidable.*

The remainder of this section is devoted to the proof of this theorem and to an example of bounded satisfiability checking. The decision procedure is closely related to the canonical model construction for modal logic [15].
Let φ be a given L_ν formula with k_φ as maximal constant. Let K be the set of formula clocks occurring in φ. Given C a set of clocks (with $C \cap K = \emptyset$) and M an integer, we want to decide if there exists a (C, M)–automaton satisfying φ. Let $C^+ = C \cup K$. Let L^φ_ν be the set of all subformulae of φ. Obviously L^φ_ν is finite.

A *problem* Π is a subset of $\mathcal{R}^{C^+}_k \times L^\varphi_\nu$ where $k = max(M, k_\varphi)$. A problem Π is said to be *satisfiable* if there exists a (C, M)-automaton A and a node η of A such that for any $(\gamma, \psi) \in \Pi$ we have $[\eta, \gamma]_{A^+} \models_{\mathcal{D}} \psi$. We call A a solution to Π. A problem Π is said to be *maximal* if it satisfies the classical closure conditions for the boolean operators and the following ones: $(\gamma, \exists\psi) \in \Pi \;\Rightarrow\; \exists l.\,(\gamma^l, \psi) \in \Pi$; $(\gamma, \forall\psi) \in \Pi \;\Rightarrow\; \forall l.\,(\gamma^l, \psi) \in \Pi$; $(\gamma, x \;\mathsf{in}\; \psi) \in \Pi \;\Rightarrow\; ([\{x\} \to 0]\gamma, \psi) \in \Pi$;

We have the two following remarks, the proofs of which are trivial: (1) If $\Pi \subseteq \Pi'$ and Π' is satisfiable then also Π is satisfiable, and (2) If Π is satisfiable then there exists a maximal problem Π' containing Π and being satisfiable.

Thus it suffices to consider satisfiability of maximal problems. Given a problem Π, a region γ and an action a we define the problem $\Pi_a^{\gamma,r}$ as the set $\{(r(\gamma), \psi) \;\mid\; (\gamma, [a]\,\psi) \in \Pi\}$. Now we introduce a new notion about problems. Let $\mathcal{C}$ be a set of maximal problems. Then $\mathcal{C}$ is a *consistency relation* if whenever $\Pi \in \mathcal{C}$ then:

1– $(\gamma, x + m \bowtie y + n) \in \Pi \;\Rightarrow\; \gamma(x) + m \bowtie \gamma(y) + n$
2– $\forall\gamma,\ (\gamma, \mathtt{ff}) \notin \Pi$
3– $(\gamma, \langle a\rangle\,\psi) \in \Pi \;\Rightarrow\; \exists\, r \subseteq C, b \in \mathcal{B}_M(C)$ and $\Pi' \in \mathcal{C}$ **s.t. :** $b(\gamma) = \mathtt{tt} \,\wedge$
$\quad (\,(r(\gamma), \psi) \,\cup\, \Pi_a^{\gamma,r} \;\subseteq\; \Pi'\,) \,\wedge\, (\forall\gamma',\, b(\gamma') = \mathtt{tt} \;\Rightarrow\; \Pi_a^{\gamma',r} \subseteq \Pi')$

We say that a maximal problem is consistent if it belongs to some consistency relation. We have the following key lemma:

Lemma 7. *Let Π be a maximal problem. Then Π is consistent if and only if Π is satisfiable.*

Proof. $\Rightarrow$ Let $\mathcal{C}$ be a consistency relation (containing Π). Now construct the canonical automaton $A_{\mathcal{C}} = \langle \mathcal{A}, N, \eta_0, C, E\rangle$ s.t. : $N = \{\eta_\Pi \mid \Pi \in \mathcal{C}\}$, η_0 is some $\eta_\Pi \in N$, and $\langle \eta_\Pi, \eta_{\Pi'}, a, r, b\rangle \in E$ iff whenever $(\gamma, [a]\,\psi) \in \Pi$ and $b(\gamma) = \mathtt{tt}$ then $(r(\gamma), \psi) \in \Pi'$.

Now it can be shown that $A_{\mathcal{C}}$ solves all problems of $\mathcal{C}$. In particular whenever $(\gamma, \psi) \in \Pi$ for some $\Pi \in \mathcal{C}$, then $[\eta_\Pi, \gamma]_{A^+_{\mathcal{C}}} \models_{\mathcal{D}} \psi$. Finally we have:

Lemma 8. *It is decidable whether a maximal problem is consistent.*

Proof. Let S_{Π_m} be the set of maximal problems over $\mathcal{R}_k^{C^+} \times L_\nu^\varphi$. Clearly S_{Π_m} is finite (since L_ν^φ and $\mathcal{R}_k^{C^+}$ are too). Thus the set of relations $\mathcal{C}$ over maximal problems is finite. Now given a relation $\mathcal{C}$ it is easy to check whether $\mathcal{C}$ is consistent since the choices for possible reset set r over C and the set $\mathcal{B}_M(C)$ are both finite.

Thus given a formula φ and bounds C and M, we can consider the (finitely many) maximal problems Π over C and M containing (γ_0, φ). It follows that φ is (C, M)–satisfiable precisely if one of these maximal problems is consistent, which is decidable due to Lemma 8. Note that the proof of Lemma 7 is constructive: given a consistency relation it gives a (C, M)-timed automata satisfying φ.

Example 5. Consider the formula φ in Example 2: We can use the model construction algorithm presented above to show that no $(1, 1)$-automaton satisfies φ.

Thus the formula in the above example is satisfiable by a 2–clock automaton but by no $(1,1)$–automata. Using the easily established fact that timed bisimilar automata satisfy the same L_ν–formulas it follows that the automaton of Example 2 is inequivalent to all $(1,1)$–automata with respect to timed bisimilarity. Now combining the above bounded model–construction algorithm with the characteristic property construction of the previous section we obtain an algorithm for deciding whether a timed automaton can be simplified in either its number clocks or the size of the constants these clocks are compared to: given a timed automaton A, a clock set C and a natural number M, it is decidable whether there exists a (C, M)–automaton being timed bisimilar to A.

Conclusion

This paper has presented two main results relating timed automata and the real–timed logic L_ν: a *characteristic formula* construction, and, a *bounded model construction* algorithm. The results presented may be pursued and improved in a number of directions: The notion of a characteristic formula construction may be applied to other behavioural preorders. In related work, we have already shown that characteristic formula constructs also exists for the "faster–than"–relation in [11] and the time–abstracted equivalence in [19]; The results of this paper only settles decidability of a *bounded* satisfiability problem for L_ν. However, it follows from this result that the unconstrained satisfiability problem is at least r.e. Decidability of the satisfiability problem with only bounds on the number of clocks remains an open problem. Finally, future work includes study of the decidability of the satisfiability problems for extensions of L_ν.

References

1. R. Alur, C. Courcoubetis, and D. Dill. Model–checking for Real–Time Systems. In *Proceedings of Logic in Computer Science*, pages 414–425. IEEE Computer Society Press, 1990.
2. R. Alur and D. Dill. Automata for Modelling Real–Time Systems. *Theoretical Computer Science*, 126(2):183–236, April 1994.
3. H.R. Andersen. Model checking and boolean graphs. In *Proceedings of ESOP'92*, volume 582 of *Lecture Notes in Computer Science, Springer Verlag*, Berlin, 1992. Springer.
4. A. Arnold and P. Crubille. A linear algorithm to solve fixed–point equations on transition systems. *Information Processing Letters*, 29, 1988.
5. M. C. Browne, E. M. Clarke, and O. Grümberg. Characterizing finite Kripke structures in propositional temporal logic. *Theoretical Computer Science*, 59:115–131, 1988.
6. Karlis Cerans. Decidability of bisimulation equivalences for parallel timer processes. In *Proc. of CAV'92*, volume 663 of *Lecture Notes in Computer Science, Springer Verlag*, Berlin, 1992. Springer Verlag.
7. E. M. Clarke and E. A. Emerson. Design and synthesis of synchronization skeletons using Branching Time Temporal Logic. In *Proc. Workshop on Logics of*

Programs, volume 131 of *Lecture Notes in Computer Science*, pages 52–71, Berlin, 1981. Springer Verlag.

8. E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite state concurrent system using temporal logic. *ACM Trans. on Programming Languages and Systems*, 8(2):244–263, 1986.
9. R. Cleaveland and B. Steffen. Computing behavioural relations, logically. In *Proceedings of 18th International Colloquium on Automata, Languages and Programming*, volume 510 of *Lecture Notes in Computer Science, Springer Verlag*, Berlin, 1991. Springer.
10. E.A. Emerson and C.L Lei. Efficient model checking in fragments of the propositional mu–calculus. In *Proceedings of Logic in Computer Science*, pages 267–278. IEEE Computer Society Press, 1986.
11. F.Moller and C. Tofts. Relating Processes with Respect to Speed. Technical Report ECS–LFCS–91–143, Department of Computer Science, University of Edinburgh, 1991.
12. S. Graf and J. Sifakis. A Modal Characterization of Observational Congruence on Finite Terms of CCS. *Information and Control*, 68:125–145, 1986.
13. T. A. Henzinger, Z. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. In *Logic in Computer Science*, 1992.
14. U. Holmer, K.G. Larsen, and W. Yi. Decidability of bisimulation equivalence between regular timed processes. In *Proceedings of CAV'91*, volume 575 of *Lecture Notes in Computer Science, Springer Verlag*, Berlin, 1992.
15. G.E. Hughes and M.J. Cresswell. *An Introduction to Modal Logic.* Methuen and Co., 1968.
16. A. Ingolfsdottir and B. Steffen. Characteristic formulae. *Information and Computation*, 110(1), 1994. To appear.
17. D. Kozen. Results on the propositional mu–calculus. In *Proc. of International Colloquium on Algorithms, Languages and Programming 1982*, volume 140 of *Lecture Notes in Computer Science, Springer Verlag*, Berlin, 1982.
18. F. Laroussinie, K. G. Larsen, and C. Weise. From Timed Automata to Logic — and Back. Technical Report RS–95–2, BRICS, 1995. Accessible through WWW: http://www.brics.aau.dk/BRICS.
19. K.G. Larsen and Y. Wang. Time Abstracted Bisimulation: Implicit Specifications and Decidability. *In Proceedings of MFPS'93*, 1993.
20. R. Milner. *Communication and Concurrency.* prentice, Englewood Cliffs, 1989.
21. D. Park. Concurrency and automata on infinite sequences. In *Proceedings of 5th GI Conference*, volume 104 of *Lecture Notes in Computer Science, Springer Verlag*, Berlin, 1981. Springer.
22. J. P. Queille and J. Sifakis. Specification and verification of concurrent programs in CESAR. In *Proc. 5th Internat. Symp. on Programming*, volume 137 of *Lecture Notes in Computer Science*, pages 195–220, Berlin, 1982. Springer Verlag.
23. A. Tarski. A lattice–theoretical fixpoint theorem and its applications. *Pacific Journal of Math.*, 5, 1955.
24. Liu Xinxin. *Specification and Decomposition in Concurrency.* PhD thesis, Aalborg University, 1992. R 92–2005.