Timed Automata Can Always Be Made Implementable*

Patricia Bouyer[1], Kim G. Larsen[2], Nicolas Markey[1], Ocan Sankur[1], and Claus Thrane[2]

[1] LSV, CNRS & ENS Cachan, France.
{bouyer,markey,sankur}@lsv.ens-cachan.fr

[2] Dept. Computer Science, Aalborg University, Denmark.
{kgl,crt}@cs.aau.dk

Abstract. Timed automata follow a mathematical semantics, which assumes perfect precision and synchrony of clocks. Since this hypothesis does not hold in digital systems, properties proven formally on a timed automaton may be lost at implementation. In order to ensure implementability, several approaches have been considered, corresponding to different hypotheses on the implementation platform. We address two of these: A timed automaton is samplable if its semantics is preserved under a discretization of time; it is robust if its semantics is preserved when all timing constraints are relaxed by some small positive parameter.
We propose a construction which makes timed automata implementable in the above sense: From any timed automaton $\mathcal{A}$, we build a timed automaton $\mathcal{A}'$ that exhibits the same behaviour as $\mathcal{A}$, and moreover $\mathcal{A}'$ is both robust and samplable by construction.

1 Introduction

Timed automata [3] extend finite-state automata with real-valued variables which measure delays between actions. They provide a powerful yet natural way of modelling real-time systems. They also enjoy decidability of several important problems, which makes them a model of choice for the verification of real-time systems. This has been witnessed over the last twenty years by substantial effort from the verification community to equip timed automata with efficient tool support, which was accompanied by successful applications.

However, timed automata are governed by a mathematical semantics, which assumes continuous and infinitely precise measurement of time, while hardware is digital and imprecise. Hence properties proven at the formal level might be lost when implementing the abstract model of the automaton as a digital circuit or as a program on a physical CPU. Several approaches have been proposed to overcome this discrepancy, with different hypotheses on the implementation

* This work has been partly supported by EU FP7 project Quasimodo (ICT-214755), and by French ANR projects DOTS (ANR-06-SETI-003) and ImpRo (ANR-10-BLAN-0317).

platform (*e.g.* [4, 15, 20, 12, 5, 21]). In this work, we address two such approaches, namely, the sampled semantics and the robustness, which we now detail.

Sampled semantics for timed automata, where all time delays are integer multiples of a rational sampling rate, have been studied in order to capture, for example the behaviour of digital circuits (*e.g.* [4, 8]). In fact, only such instants are observable in a digital circuit, under the timing of a quartz clock. However, for some timed automata, any sampling rate may disable some (possibly required) behaviour [9]. Consequently, a natural problem which has been studied is that of choosing a sampling rate under which a property is satisfied. For safety properties, this problem is undecidable for timed automata [9]; but it becomes decidable for reachability under a slightly different setting [17]. Recently, [1] showed the decidability of the existence of a sampling rate under which the continuous and the sampled semantics recognize the same untimed language.

A prominent approach, originating from [20, 12], for verifying the behavior of real-time programs executed on CPUs, is robust model-checking. It consists in studying the *enlarged semantics* of the timed automaton, where all the constraints are enlarged by a small (positive) perturbation Δ, in order to model the imprecisions of the clock. In some cases [11], this may allow new behaviours in the system, regardless of Δ (See Fig. 2 on page 8). Such automata are said to be not *robust* to small perturbations. On the other hand, if no new behaviour is added to the system, that is, if the system is robust, then *implementability* on a fast-enough CPU will be ensured [12]. Since its introduction, robust model-checking has been solved for safety properties [20, 11], and for richer linear-time properties [6, 7]. See also [21] for a variant of the implementation model of [12] and a new approach to obtain implementations.

In this paper, we show that timed automata can always be made implementable in both senses. More precisely, given a timed automaton $\mathcal{A}$, we build another timed automaton $\mathcal{B}$ whose semantics under enlargement and under sampling is bisimilar to $\mathcal{A}$. We use a quantitative variant of bisimulation from [14] where the differences between the timings in two systems are bounded above by a parameter ε (see also [16] for a similar quantitative notion of bisimulation). Our construction is parameterized and provides a bisimilar implementation for any desired precision $\varepsilon > 0$. Moreover, we prove that in timed automata, this notion of bisimulation preserves, up to an error of ε, all properties expressed in a quantitative extension of CTL, also studied in [14].

2 Timed Models and Specifications

2.1 Timed Transition Systems and Behavioural Relations

A *timed transition system (TTS)* is a tuple $\mathcal{S} = (S, s_0, \Sigma, \mathbb{K}, \to)$, where S is the set of *states*, $s_0 \in S$ the initial state, Σ a finite alphabet, $\mathbb{K} \subseteq \mathbb{R}_{\geq 0}$ the time domain which contains 0 and is closed under addition, and $\to \subseteq S \times (\Sigma \cup \mathbb{K}) \times S$ the *transition* relation. We write $s \xrightarrow{\sigma} s'$ instead of $(s, \sigma, s') \in \to$; we also write $s \xrightarrow{d,\sigma} s'$ if $s \xrightarrow{d} s'' \xrightarrow{\sigma} s'$ for $d \in \mathbb{K}$, $\sigma \in \Sigma$ and some state s'', and $s \stackrel{\sigma}{\Longrightarrow} s'$

if $s \xrightarrow{d',\sigma} s'$ for some $d' \in \mathbb{K}$. A *run* ρ of $\mathcal{S}$ is a finite or infinite sequence $q_0 \xrightarrow{\tau_0} q_0' \xrightarrow{\sigma_0} q_1 \xrightarrow{\tau_1} q_1' \xrightarrow{\sigma_1} \ldots$, where $q_i \in S$, $\sigma_i \in \Sigma$ and $\tau_i \in \mathbb{K}$ for all i. The word $\sigma_0\sigma_1\ldots \in \Sigma^*$ is the *trace* of ρ. We denote by $\mathsf{Trace}(\mathcal{S})$ the set of finite and infinite traces of the runs of $\mathcal{S}$. We define the set of *reachable states* of $\mathcal{S}$, denoted by $\mathsf{Reach}(\mathcal{S})$, as the set of states s' for which some finite run of $\mathcal{S}$ starts from state s_0 and ends in state s'. A run written on the form $\gamma = q_0 \xrightarrow{d_0,\sigma_0} q_1 \xrightarrow{d_1,\sigma_1} q_2 \ldots$ is a *timed-action path* (or simply path). Each state $q_0 \in S$ admits a set $P(q_0)$ of paths starting at q_0. For any path γ, the suffix γ^j is obtained by deleting the first j transitions in γ, and $\gamma(j) = q_j \xrightarrow{d_j,\sigma_j} q_{j+1}$ is the j-th transition in γ; we also let $\mathsf{state}_j(\gamma) = q_j$, $\gamma(j)_\sigma = \sigma_j$, and $\gamma(j)_d = d_j$.

We consider a quantitative extension of timed bisimilarity introduced in [22]. This spans the gap between timed and time-abstract bisimulations: while the former requires time delays to be matched exactly, the latter ignores timing information altogether. Intuitively, we define two states to be ε-bisimilar, for a given parameter $\varepsilon \geq 0$, if there is a (time-abstract) bisimulation which relates these states in such a way that, at each step, the difference between the time delays of corresponding delay transitions is at most ε. Thus, this parameter allows one to quantify the "timing error" made during the bisimulation. A strong and a weak variant of this notion is given in the following definition.

Definition 1. *Given a TTS $(S, s_0, \Sigma, \mathbb{K}, \to)$, and $\varepsilon \geq 0$, a symmetric relation $R_\varepsilon \subseteq S \times S$ is a*

- strong timed ε-bisimulation*, if for any $(s,t) \in R_\varepsilon$ and $\sigma \in \Sigma, d \in \mathbb{K}$,*
  - *$s \xrightarrow{\sigma} s'$ implies $t \xrightarrow{\sigma} t'$ for some $t' \in S$ with $(s',t') \in R_\varepsilon$,*
  - *$s \xrightarrow{d} s'$ implies $t \xrightarrow{d'} t'$ for some $t' \in S$ and $d' \in \mathbb{K}$ with $|d - d'| \leq \varepsilon$ and $(s',t') \in R_\varepsilon$.*
- timed-action ε-bisimulation*, if for any $(s,t) \in R_\varepsilon$, and $\sigma \in \Sigma$, $d \in \mathbb{K}$,*
  - *$s \xrightarrow{d,\sigma} s'$ implies $t \xrightarrow{d',\sigma} t'$ for some $t' \in S$ and $d' \in \mathbb{K}$ with $|d - d'| \leq \varepsilon$ and $(s',t') \in R_\varepsilon$.*

If there exists a strong timed ε-bisimulation (resp. timed-action ε-bisimulation) R_ε such that $(s,t) \in R_\varepsilon$, then we write $s \sim_\varepsilon t$ (resp. $s \approx_\varepsilon t$). Furthermore we write $s \sim_{\varepsilon^+} t$ (resp. $s \approx_{\varepsilon^+} t$) whenever for every $\varepsilon' > \varepsilon$, $s \sim_{\varepsilon'} t$ (resp. $s \approx_{\varepsilon'} t$).

Observe that $s \sim_\varepsilon t$ implies $s \sim_{\varepsilon'} t$ for every $\varepsilon' > \varepsilon$. Also, $s \sim_{\varepsilon^+} t$ does not imply $s \sim_\varepsilon t$ in general (see Fig. 1), and if $s \sim_{\varepsilon^+} t$ but $s \not\sim_\varepsilon t$, then $\varepsilon = \inf\{\varepsilon' > 0 \mid s \sim_{\varepsilon'} t\}$. These observations hold true in the timed-action bisimulation setting as well. Note also that $s \sim_\varepsilon t$ implies $s \approx_\varepsilon t$. Finally, for $\varepsilon > 0$, strong timed or timed-action ε-bisimilarity relations are not equivalence relations in general, but they are when $\varepsilon = 0$.

Last, we define a variant of *ready-simulation* [18] for timed transition systems. For $\mathsf{Bad} \subseteq \Sigma$, we will write $I \sqsubseteq^{\mathsf{Bad}} S$ when I is simulated by S (and time delays are matched exactly) in such a way that at any time during the simulation, any failure (*i.e.*, any action in Bad) enabled in S is also enabled in I. So, if $I \sqsubseteq^{\mathsf{Bad}} S$ and S is safe w.r.t. Bad (*i.e.*, Bad actions are never enabled), then any

Fig. 1. An automaton in which $(s,0) \sim_{0^+} (t,0)$ but $(s,0) \not\sim_0 (t,0)$.

run of I can be executed in S (with exact timings) without enabling any of the Bad-actions. Fig. 2 will provide an automaton illustrating the importance of this notion. More formally:

Definition 2. *Given a TTS $(S, s_0, \Sigma, \mathbb{K}, \to)$, and a set $\mathsf{Bad} \subseteq \Sigma$, a relation $R \subseteq S \times S$ is a* ready-simulation w.r.t. Bad *if, whenever $(s,t) \in R$:*

- *for all $\sigma \in \Sigma$ and $d \in \mathbb{K}$, $s \xrightarrow{d,\sigma} s'$ implies $t \xrightarrow{d,\sigma} t'$ for some $t' \in S$ with $(s',t') \in R$,*
- *for all $\sigma \in \mathsf{Bad}$, $t \overset{\sigma}{\Longrightarrow} t'$ implies $s \overset{\sigma}{\Longrightarrow} s'$ for some $s' \in S$.*

We write $s \sqsubseteq^{\mathsf{Bad}} t$ if $(s,t) \in R$ for some ready-simulation R w.r.t. Bad.

2.2 Timed Automata

Given a set of clocks $\mathcal{C}$, the elements of $\mathbb{R}^{\mathcal{C}}_{\geq 0}$ are referred to as *valuations*. For a subset $X \subseteq \mathcal{C}$, and a valuation v, we define $v[X \leftarrow 0]$ as the valuation $v[X \leftarrow 0](x) = v(x)$ for all $x \in \mathcal{C} \setminus X$ and $v[X \leftarrow 0](x) = 0$ for $x \in X$. For any $d \in \mathbb{R}_{\geq 0}$, $v + d$ is the valuation defined by $(v+d)(x) = v(x) + d$ for all $x \in \mathcal{C}$. For any $\alpha \in \mathbb{R}$, we define αv as the valuation obtained by multiplying all components of v by α, that is $(\alpha v)(x) = \alpha v(x)$ for all $x \in \mathcal{C}$. Given two valuations v and v', we denote by $v + v'$ the valuation that is the componentwise sum of v and v', that is $(v+v')(x) = v(x) + v'(x)$ for all $x \in \mathcal{C}$.

Let $\mathbb{Q}_\infty = \mathbb{Q} \cup \{-\infty, \infty\}$. An *atomic clock constraint* is a formula of the form $k \preceq x \preceq' l$ or $k \preceq x - y \preceq' l$ where $x, y \in \mathcal{C}$, $k, l \in \mathbb{Q}_{\geq 0}$ and $\preceq, \preceq' \in \{<, \leq\}$. A *guard* is a conjunction of atomic clock constraints. For $M, \eta \in \mathbb{Q}_{>0}$ such that $\frac{1}{\eta} \in \mathbb{N}$, we denote by $\Phi_{\mathcal{C}}(\eta, M)$ the set of guards on the clock set $\mathcal{C}$, whose constants are either $\pm\infty$ or less than or equal to M in absolute value and are integer multiples of η. Let $\Phi_{\mathcal{C}}$ denote the set of all guards on clock set $\mathcal{C}$. A valuation v satisfies $\varphi \in \Phi_{\mathcal{C}}$ if all atomic clock constraints of φ are satisfied when each $x \in \mathcal{C}$ is replaced by $v(x)$. Let $[\![\varphi]\!]$ denote the set of valuations that satisfy φ. We define the *enlargement* of atomic clock constraints by $\Delta \in \mathbb{Q}$ as

$$\langle k \preceq x - y \preceq' l \rangle_\Delta = k - \Delta \preceq x - y \preceq' l + \Delta,$$
$$\text{and} \qquad \langle k \preceq x \preceq' l \rangle_\Delta = k - \Delta \preceq x \preceq' l + \Delta.$$

for $x, y \in \mathcal{C}$ and $k, l \in \mathbb{Q}_{>0}$. The enlargement of a guard φ, denoted by $\langle \varphi \rangle_\Delta$, is obtained by enlarging all its atomic clock constraints.

Definition 3. *A timed automaton $\mathcal{A}$ is a tuple $(\mathcal{L}, \mathcal{C}, \Sigma, l_0, E)$, consisting of a finite set $\mathcal{L}$ of locations, a finite set $\mathcal{C}$ of clocks, a finite alphabet Σ of labels, a finite set $E \subseteq \mathcal{L} \times \Phi_{\mathcal{C}} \times \Sigma \times 2^{\mathcal{C}} \times \mathcal{L}$ of edges, and an initial location $l_0 \in \mathcal{L}$.*

We write $l \xrightarrow{\varphi,\sigma,R} l'$ *if* $e = (l, \varphi, \sigma, R, l') \in E$*, and call* φ *the guard of* e*.* $\mathcal{A}$ *is an* integral *timed automaton if all constants that appear in its guards are integers.*

We call the inverses of positive integers *granularities*. The *granularity* of a timed automaton is the inverse of the least common denominator of the finite constants in its guards. For any timed automaton $\mathcal{A}$ and rational $\Delta \geq 0$, let $\mathcal{A}_\Delta$ denote the timed automaton obtained from $\mathcal{A}$ where each guard φ is replaced with $\langle\varphi\rangle_\Delta$.

Definition 4. *The semantics of a timed automaton* $\mathcal{A} = (\mathcal{L}, \mathcal{C}, \Sigma, l_0, E)$ *is a TTS over alphabet* Σ*, denoted* $[\![\mathcal{A}]\!]$*, whose state space is* $\mathcal{L} \times \mathbb{R}^{\mathcal{C}}_{\geq 0}$*. The initial state is* $(l_0, \mathbf{0})$*, where* $\mathbf{0}$ *denotes the valuation where all clocks have value* 0. Delay transitions *are defined as* $(l, v) \xrightarrow{\tau} (l, v+\tau)$ *for any state* (l, v) *and* $\tau \in \mathbb{K}$. Action transitions *are defined as* $(l, v) \xrightarrow{\sigma} (l', v')$*, for any edge* $l \xrightarrow{g,\sigma,R} l'$ *in* $\mathcal{A}$ *such that* $v \models g$ *and* $v' = v[R \leftarrow 0]$.

For any $k \in \mathbb{N}_{>0}$*, we define the* sampled semantics *of* $\mathcal{A}$*, denoted by* $[\![\mathcal{A}]\!]^{\frac{1}{k}}$ *as the TTS defined similarly to* $[\![\mathcal{A}]\!]$ *by taking the time domain as* $\mathbb{K} = \frac{1}{k}\mathbb{N}$.

We write $[\![\mathcal{A}]\!] \sim_\varepsilon [\![\mathcal{B}]\!]$, $[\![\mathcal{A}]\!] \approx_\varepsilon [\![\mathcal{B}]\!]$ and $[\![\mathcal{A}]\!] \sqsubseteq^{\mathsf{Bad}} [\![\mathcal{B}]\!]$ when the initial states of timed automata $\mathcal{A}$ and $\mathcal{B}$ are related accordingly in the disjoint union of the transition systems, defined in the usual way.

We define the usual notion of region equivalence [3]. Let M be the maximum (rational) constant that appears in the guards of $\mathcal{A}$, let η be the granularity of $\mathcal{A}$. Multiplying any constant in $\mathcal{A}$ by $\frac{1}{\eta}$, we obtain an integral timed automaton.

Given valuations $u, v \in \mathbb{R}^{\mathcal{C}}_{\geq 0}$ and rationals M, η, define $v \simeq^M_\eta u$ to hold if, and only if, for all formulas $\varphi \in \Phi_{\mathcal{C}}(\eta, M)$, $u \models \varphi$ if and only if $v \models \varphi$. The equivalence class of a valuation u for the relation $\simeq^M_\eta$ is denoted by $\mathsf{reg}(u)^M_\eta = \{v \mid u \simeq^M_\eta v\}$. Each such class is called an (η, M)-region. In the rest, when constant M is (resp. M and η are) clear from context, we simply write $\mathsf{reg}(u)_\eta$ (resp. $\mathsf{reg}(u)$) and call these η-regions (resp. regions). We denote by $\overline{\mathsf{reg}(u)^M_\eta}$ the topological closure of $\mathsf{reg}(u)^M_\eta$. The number of (η, M)-regions is bounded by $O(2^{|\mathcal{C}|}|\mathcal{C}|!(M/\eta)^{|\mathcal{C}|})$ [3].

For a region r, we denote by $r[R \leftarrow 0]$, the region obtained by resetting clocks in R. We define $\mathsf{tsucc}^*(r)$ as the set of *time-successor regions of* r, that is, the set of η-regions r' such that $u + d \in r'$ for some $u \in r$ and $d \in \mathbb{R}_{\geq 0}$.

We now associate with each (η, M)-region a guard that defines it. Assume we number the clocks with indices so that $\mathcal{C} = \{x_1, \ldots, x_m\}$, and fix any (η, M)-region r. Let us define $x_0 = 0$, and $\mathcal{C}_0 = \mathcal{C} \cup \{x_0\}$. Then, for each pair $i, j \in \mathcal{C}_0$, there exists a number $A_{i,j} \in \eta\mathbb{Z} \cap [-M, M] \cup \{\infty\}$ and $\prec_{i,j} \in \{<, \leq\}$ s.t. φ_r, defined as

$$\varphi_r = \bigwedge_{(x_i, x_j) \in \mathcal{C}_0} -A_{j,i} \prec_{j,i} x_i - x_j \prec_{i,j} A_{i,j},$$

is such that $[\![\varphi_r]\!] = r$. Moreover, we assume that for all $i, j, k \in \mathcal{C}_0$, $A_{i,i} = 0$ and $A_{i,j} \leq A_{i,k} + A_{k,j}$. Note that this is a standard definition: the matrix $(A_{i,j})_{i,j}$ is a *difference-bound matrix (DBM)* that defines region r, and the latter condition defines its *canonical form* [13]. Later we will refer to matrix $(A_{i,j})_{i,j}$ as the DBM that defines region r.

2.3 Quantitative Extension of Computation Tree Logic

In the style of [10, 16, 14] we present a quantitative extension of CTL, which measures (in a sense that we make clear below) how far a formula is from being satisfied in a given state.

Definition 5. *Let $\mathbb{I}$ be the set of closed nonempty intervals of $\mathbb{R}_{\geq 0}$, and Σ be a finite alphabet. We define the set of state- and path-formulas as follows*[1]

$$\Psi ::= \top \mid \bot \mid \Psi_1 \wedge \Psi_2 \mid \Psi_1 \vee \Psi_2 \mid \mathsf{E}\Pi \mid \mathsf{A}\Pi$$
$$\Pi ::= \mathsf{X}_A^I \Psi \mid \bar{\mathsf{X}}_A^I \Psi \mid \Psi_1 \mathsf{R}^I \Psi_2 \mid \Psi_1 \mathsf{U}^I \Psi_2$$

for $I \in \mathbb{I}$ and $A \subseteq \Sigma$. We write $\mathcal{L}_T(\Sigma)$ or simply $\mathcal{L}_T$ for the set of state formulae.

To define the semantics of $\mathcal{L}_T$, we introduce the distance between a point and an interval: $|z, [x, y]| = 0$ when $z \in [x, y]$, and $|z, [x, y]| = \min\{|x - z|, |y - z|\}$ otherwise. Now, given a state s, the value of a state formula is defined inductively as follows: $(\!|\top|\!)(s) = 0$, $(\!|\bot|\!)(s) = \infty$, and

$$(\!|\psi_1 \vee \psi_2|\!)(s) = \inf\left\{(\!|\psi_1|\!)(s), (\!|\psi_2|\!)(s)\right\} \qquad (\!|\mathsf{E}\pi|\!)(s) = \inf\left\{(\!|\pi|\!)(\gamma) \mid \gamma \in P(s)\right\}$$
$$(\!|\psi_1 \wedge \psi_2|\!)(s) = \sup\left\{(\!|\psi_1|\!)(s), (\!|\psi_2|\!)(s)\right\} \qquad (\!|\mathsf{A}\pi|\!)(s) = \sup\left\{(\!|\pi|\!)(\gamma) \mid \gamma \in P(s)\right\}$$

For a path γ, it is defined as:

$$(\!|\mathsf{X}_A^I \psi|\!)(\gamma) = (\!|\bar{\mathsf{X}}_A^I \psi|\!)(\gamma) = \max\{|\gamma(0)_d, I|, (\!|\psi|\!)(\mathsf{state}_1(\gamma))\} \qquad \text{if } \gamma(0)_\sigma \in A$$
$$(\!|\mathsf{X}_A^I \psi|\!)(\gamma) = +\infty \quad \text{and} \quad (\!|\bar{\mathsf{X}}_A^I \psi|\!)(\gamma) = 0 \qquad \text{if } \gamma(0)_\sigma \notin A$$
$$(\!|\psi_1 \mathsf{U}^I \psi_2|\!)(\gamma) = \inf_k \left(\max\left\{ \max_{0 \leq j < k} |(\!|\psi_1|\!)(\mathsf{state}_j(\gamma)), I|,\ (\!|\psi_2|\!)(\mathsf{state}_k(\gamma))\right\}\right)$$
$$(\!|\psi_1 \mathsf{R}^I \psi_2|\!)(\gamma) = \sup_k \left(\min\left\{ \max_{0 \leq j < k} |(\!|\psi_1|\!)(\mathsf{state}_j(\gamma)), I|, (\!|\psi_2|\!)(\mathsf{state}_k(\gamma))\right\}\right)$$

For instance, $(\!|\mathsf{EX}_{\{a\}}^{[2,5]} \top|\!)(s)$ is the lower bound of the set

$$\{\left|d, [2, 5]\right| \mid \text{there is a transition } s \xrightarrow{d,a} s'\}.$$

Intuitively, this semantics measures the amount of point-wise modifications (in the timing constraints of the formula) that are needed for this formula to hold at a given state. Notice that untimed[2] formulas of $\mathcal{L}_T$ can only be evaluated to 0 or $+\infty$, and this value reflects the Boolean value of the underlying CTL formula.

It is shown in [22] that $\mathcal{L}_T$ characterizes ε-bisimilarity between the states of *weighted Kripke structures*. In the following proposition, we generalize one direction of this result to timed automata, showing that ε-bisimilar states have close satisfaction values for all formulas of $\mathcal{L}_T$, which implies that these properties (and their values) are preserved upto ε by the constructions we give in Section 4.

[1] To establish the relationship to timed-action ε-bisimulation, the logic uses actions on transitions instead of the more usual atomic propositions on states. It is easy to encode the latter by adding extra transitions to sink states.

[2] *I.e.*, when all timing constraints are $[0, +\infty)$.

Proposition 1. *For any timed automaton $\mathcal{A}$ and states s, s' of $[\![\mathcal{A}]\!]$, for all $\varepsilon \geq 0$, if $s \approx_{\varepsilon+} s'$ then for any $\psi \in \mathcal{L}_T$ either $(\!|\psi|\!)(s) = (\!|\psi|\!)(s') = \infty$ or $|(\!|\psi|\!)(s) - (\!|\psi|\!)(s')| \leq \varepsilon$.*

3 Implementability

As explained in the introduction, even the smallest enlargement of the guards may yield extra behaviour in timed automata. Similarly, any sampling of the time domain may remove behaviours. Here, we give several definitions of *robustness* and *samplability*, which distinguish timed automata whose enlargement (resp. whose sampled semantics) is ε-bisimilar to the original automaton, for some ε.

3.1 Robustness

Earlier work on robustness based on enlargement, such as [11, 6, 7] concentrated on deciding the existence of a positive Δ under which the enlarged automaton is correct w.r.t. a given property. Here, we consider a stronger notion of robustness, which requires systems to be ε-bisimilar for some ε.

Definition 6. *A timed automaton $\mathcal{A}$ is* ε-bisimulation-robust *(or simply ε-robust), where $\varepsilon \geq 0$, if there exists $\Delta > 0$ such that $[\![\mathcal{A}]\!] \approx_\varepsilon [\![\mathcal{A}_\Delta]\!]$.*

Note that not all timed automata are robust. In fact, in the automaton $\mathcal{A}$ of Fig. 2, location ℓ_3 is not reachable in $[\![\mathcal{A}]\!]$, but it becomes reachable in $[\![\mathcal{A}_\Delta]\!]$ for any $\Delta > 0$ (see [11]).

We do not know whether a timed automaton that is robust for some Δ is still robust for any $\Delta' < \Delta$, that is, whether $[\![\mathcal{A}]\!] \approx_\varepsilon [\![\mathcal{A}_\Delta]\!]$ implies $[\![\mathcal{A}]\!] \approx_\varepsilon [\![\mathcal{A}_{\Delta'}]\!]$ for $\Delta' < \Delta$, in general. This is the so-called "faster-is-better" property [2, 12], which means that if a property holds in some platform, it also holds in a faster or more precise platform. This is known to be satisfied for simpler notions of robustness mentioned above.

In the next section, we will present our construction which, for any $\mathcal{A}$, produces an alternative automaton $\mathcal{A}'$ which is robust and satisfies $[\![\mathcal{A}]\!] \approx_\varepsilon [\![\mathcal{A}'_\Delta]\!]$ for all small enough Δ.

Bisimulation is not always sufficient when one wants to preserve state-based safety properties proven for $\mathcal{A}$. For instance, removing edges leading to unsafe states in $\mathcal{A}$ may provide us with a trivially safe automaton under any enlargement. However, edges leading to such states are used to detect failures, so removing these will not necessarily remove the failure (since the states that immediately trigger a failure may still be reachable). Fig. 3 gives such an "incorrect" construction. To cope with this problem, we rely on ready-simulation and require $\mathcal{A}'$ to satisfy $[\![\mathcal{A}'_\Delta]\!] \sqsubseteq^{\mathsf{Bad}} [\![\mathcal{A}_\Delta]\!]$, where Bad are distinguished actions leading to unsafe states in $\mathcal{A}$. This means that any run of $[\![\mathcal{A}'_\Delta]\!]$ can be realized in $[\![\mathcal{A}_\Delta]\!]$, and that no Bad-action is enabled in that run in the latter (and hence no unsafe state is reached). Thus, intuitively, no state reached in $[\![\mathcal{A}'_\Delta]\!]$ corresponds to an unsafe state in $[\![\mathcal{A}_\Delta]\!]$, and in particular, if $\mathcal{A}_\Delta$ has unsafe runs (leading to unsafe

states), then these cannot be realized in $\mathcal{A}'_\Delta$. Clearly, the automaton in Fig. 3 does not satisfy this. We formalize this idea here.

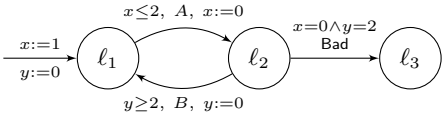

Fig. 2. A non-robust timed automaton [20].

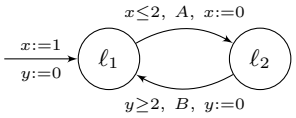

Fig. 3. A robust but unsafe alternative.

Definition 7. *A timed automaton $\mathcal{A}$ is* safe w.r.t. a set of actions $\mathsf{Bad} \subseteq \Sigma$, *if $d_\infty\big(\mathsf{Reach}(\llbracket\mathcal{A}\rrbracket), Pre(\mathsf{Bad})\big) > 0$, where d_∞ is the standard supremum metric, and $Pre(\mathsf{Bad}) = \bigcup_{\sigma\in\mathsf{Bad},(l,\sigma,g,R,l')\in E} \{l\} \times \llbracket g\rrbracket$ is the precondition of* Bad*-actions*

Notice that $Pre(\mathsf{Bad})$ is the set of states from which a Bad action can be done, and that $d_\infty(\mathsf{Reach}\llbracket\mathcal{A}\rrbracket, Pre(\mathsf{Bad})) = 0$ does not imply that a state of $Pre(\mathsf{Bad})$ is reachable in $\mathcal{A}$. But we still consider such an automaton as unsafe, since, intuitively, any enlargement of the guards may lead to a state of $Pre(\mathsf{Bad})$. It can be seen that automaton of Fig. 2 is safe w.r.t. action Bad. Note that a closed timed automaton is safe w.r.t. Bad iff Bad is not reachable.

Recall the standard notion of robustness, used *e.g.* in [11]:

Definition 8. *A timed automaton $\mathcal{A}$ is* safety-robust (w.r.t. Bad) *if there exists $\Delta > 0$ such that $\llbracket\mathcal{A}_\Delta\rrbracket$ is safe w.r.t.* Bad.

In the rest, Bad will refer to a set of actions given with the timed automaton we consider. When we say that a timed automaton is safe, or safety-robust, these actions will be implicit.

We introduce the notion of *safety-robust implementation* (parameterized by a bisimilarity relation $\equiv$, which will range over $\{\sim_0, \sim_{0^+}, \approx_0, \approx_{0^+}\}$), where we only require the alternative automaton to preserve a given safety specification.

Definition 9 (Safety-Robust Implementation). *Let $\mathcal{A}$ be a timed automaton which is safe w.r.t. actions* Bad*, and $\equiv$ denote any bisimilarity relation. A* safety-robust implementation of $\mathcal{A}$ w.r.t $\equiv$ *is a timed automaton $\mathcal{A}'$ such that:*

(*i*) *$\mathcal{A}'$ is safety-robust;*
(*ii*) *$\llbracket\mathcal{A}'\rrbracket \equiv \llbracket\mathcal{A}\rrbracket$;*
(*iii*) *there exists $\Delta_0 > 0$ s.t. for all $0 < \Delta' < \Delta < \Delta_0$, $\llbracket\mathcal{A}'_{\Delta'}\rrbracket \sqsubseteq^{\mathsf{Bad}} \llbracket\mathcal{A}_\Delta\rrbracket$.*

Now we define the notion of *robust implementation*. We require such an implementation to be robust and equivalent to the original automaton, and to preserve safety specifications.

Definition 10 (Robust Implementation). *Let $\mathcal{A}$ be a timed automaton which is safe w.r.t.actions* Bad*, and $\equiv$ denote any bisimilarity. An ε-*robust implementation of $\mathcal{A}$ w.r.t. $\equiv$ *is a timed automaton $\mathcal{A}'$ such that:*

(*i*) $\mathcal{A}'$ *is* ε*-robust;*
(*ii*) $[\![\mathcal{A}']\!] \equiv [\![\mathcal{A}]\!]$*;*
(*iii*) *there exists* $\Delta_0 > 0$ *s.t. for all* $0 < \Delta' < \Delta < \Delta_0$, $[\![\mathcal{A}'_{\Delta'}]\!] \sqsubseteq^{\mathsf{Bad}} [\![\mathcal{A}_{\Delta}]\!]$.

3.2 Samplability

As we noted in the introduction, some desired behaviours of a given timed automaton may be removed in the sampled semantics. Preservation of the untimed language under some sampling rate was shown decidable in [1]. The proof is highly technical (it is based on the limitedness problem for a special kind of counter automata).We are interested in the stronger notion of *bisimulation-samplability*, which, in particular, implies the preservation of untimed language.

Definition 11. *A timed automaton is said to be* ε-bisimulation-samplable *(or simply* ε-samplable*) if there exists a granularity* η *such that* $[\![\mathcal{A}]\!] \approx_{\varepsilon} [\![\mathcal{A}]\!]^{\eta}$.

Note that not all timed automata are bisimulation-samplable: [17] describes timed automata $\mathcal{A}$ which are not (time-abstract) bisimilar to their sampled semantics for any granularity η. We define a *sampled implementation* as follows.

Definition 12 (Sampled Implementation). *Let* $\mathcal{A}$ *be a timed automaton, and* $\equiv$ *denote any bisimilarity relation. A* ε-sampled implementation w.r.t. $\equiv$ *is a timed automaton* $\mathcal{A}'$ *such that*

(*i*) $\mathcal{A}'$ *is* ε*-samplable;*
(*ii*) $[\![\mathcal{A}']\!] \equiv [\![\mathcal{A}]\!]$.

Note that a similar phenomenon as in Fig. 2 does not occur in sampled semantics since sampling does not add extra behaviour, but may only remove some.

3.3 Main result of the paper

We will present two constructions which yield an implementation for any timed automaton. In our first construction, for any timed automaton given with a safety specification, we construct a safety robust implementation. Our second construction is stronger: Given any timed automaton $\mathcal{A}$ and any desired $\varepsilon > 0$, we construct a timed automaton $\mathcal{A}'$ which is both an ε-robust implementation and an ε-sampled implementation of $\mathcal{A}$ w.r.t. $\approx_{0^+}$ (we also give a variant w.r.t. $\sim_0$ for robustness).

Since, $\mathcal{A}$ and $\mathcal{A}'_{\Delta}$ are timed-action ε-bisimilar, the satisfaction values of the formulas in $\mathcal{L}_T$ are preserved up to ε (Proposition 1). In particular, all standard untimed linear- and branching-time properties (e.g. expressible in LTL, resp. CTL) proven for the original automaton are preserved in the implementation. An example of such a property is deadlock-freedom, which is an important property of programs.

Theorem 1. *Let* $\mathcal{A} = (\mathcal{L}, \mathcal{C}, \Sigma, l_0, E)$ *be an integral timed automaton which is safe w.r.t. some set* $\mathsf{Bad} \subseteq \Sigma$. *Let* W *denote the number of regions of* $\mathcal{A}$. *Then,*

1. *There exists a safety robust implementation of $\mathcal{A}$ w.r.t $\sim_0$, with $|\mathcal{L}|$ locations, the same number of clocks and at most $|E| \cdot W$ edges.*
2. *For all $\varepsilon > 0$, there exists a timed automaton $\mathcal{A}'$ which is a ε-robust implementation w.r.t. $\sim_0$; and a timed automaton $\mathcal{A}''$ which is both a ε-sampled and ε-robust implementation w.r.t. $\approx_{0^+}$. Both timed automata have the same number of clocks as $\mathcal{A}$, and the number of their locations and edges is bounded by $O(|\mathcal{L}| \cdot W \cdot (\frac{1}{\varepsilon})^{|\mathcal{C}|})$.*

The rest of the paper is devoted to the proof of this theorem. The two constructions are presented in the next section, and proved thereafter.

4 Making Timed Automata Robust and Samplable

For any timed automaton $\mathcal{A}$ and any location l of $\mathcal{A}$, let $\mathsf{Reach}(\llbracket\mathcal{A}\rrbracket)|_l$ denote the projection of the set of reachable states at location l to $\mathbb{R}^{\mathcal{C}}_{\geq 0}$. For any l, there exist guards $\varphi^l_1, \ldots, \varphi^l_{n_l}$ such that $\bigcup_i \llbracket\varphi^l_i\rrbracket = \mathsf{Reach}(\llbracket\mathcal{A}\rrbracket)|_l$ (in fact, the set of reachable states at a given location is a union of regions but not necessarily convex). We use these formulas to construct a new automaton where we restrict all transitions to be activated only at reachable states.

Definition 13. *Let $\mathcal{A} = (\mathcal{L}, \mathcal{C}, \Sigma, l_0, E)$ be any integral timed automaton. Define timed automaton $\mathsf{safe}(\mathcal{A})$ from $\mathcal{A}$ by replacing each edge $l \xrightarrow{\varphi,\sigma,R} l'$, by edges $l \xrightarrow{\varphi\wedge\varphi^l_i,\sigma,R} l'$ for all $i \in \{1, \ldots, n_l\}$.*

As stated in Theorem 1 the worst-case complexity of this construction is exponential. However, in practice, $\mathsf{Reach}(\llbracket\mathcal{A}\rrbracket)|_l$ may have a simple shape, which can be captured by few formulas φ^l_i.

Although the above construction will be enough to obtain a safety-robust timed automaton w.r.t. a given set Bad, it may not be bisimulation-robust. The following construction ensures this.

Definition 14. *Let $\mathcal{A} = (\mathcal{L}, \mathcal{C}, \Sigma, l_0, E)$ be an integral timed automaton. Let M be the largest constant that appears in $\mathcal{A}$, and let η be any granularity. We define $\mathsf{impl}_\eta(\mathcal{A})$ as a timed automaton over the set of locations l_r where l is a location of $\mathcal{A}$ and r is an (η, M)-region, and over the same set of clocks. Edges are defined as follows. Whenever there is an edge $l \xrightarrow{\varphi,\sigma,R} l'$ in $\mathcal{A}$, we let $l_r \xrightarrow{\varphi\wedge\varphi_s,\sigma,R} l'_{s[R\leftarrow 0]}$, for all (η, M)-regions r and $s \in \mathsf{tsucc}^*(r)$ such that $\llbracket\varphi_s\rrbracket \subseteq \llbracket\varphi\rrbracket$.*

We define $\overline{\mathsf{impl}}_\eta(\mathcal{A})$ as the closed timed automaton obtained from $\mathsf{impl}_\eta(\mathcal{A})$ where each guard is replaced by its closed counterpart[3]*.*

Throughout this paper, we always consider integral timed automata as input, and the only non-integer constants are those added by our construction. Observe that the size of $\mathsf{impl}_\eta(\mathcal{A})$ depends on η, since a smaller granularity yields a greater number of (η, M)-regions.

[3] that is, all $<$ are replaced by $\leq$, and $>$ by $\geq$.

The main theorem is a direct corollary of the following lemma, where we state our results in detail. The bounds on the size of the constructed implementations follow by construction.

Lemma 1. *Let $\mathcal{A} = (\mathcal{L}, \mathcal{C}, \Sigma, l_0, E)$ be an integral timed automaton and fix any $\varepsilon > 0$. Assume that $\mathcal{A}$ is safe w.r.t. some set $\mathsf{Bad} \subseteq \Sigma$. Then,*

1. *$\mathsf{safe}(\mathcal{A})$ is safety-robust, $[\![\mathcal{A}]\!] \sim_0 [\![\mathsf{safe}(\mathcal{A})]\!]$ and for any $\Delta < \frac{1}{2|\mathcal{C}|}$, $[\![\mathsf{safe}(\mathcal{A})_\Delta]\!] \sqsubseteq^{\mathsf{Bad}} [\![\mathcal{A}_\Delta]\!]$.*
2. *For any granularity η and $\Delta > 0$ such that $2(\eta + \Delta) < \varepsilon$, we have $[\![\mathcal{A}]\!] \approx_{0+} [\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]$ and $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!] \approx_\varepsilon [\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_\Delta]\!]$. Moreover, for any $0 < \Delta' < \Delta < \frac{1}{|\mathcal{C}|}$, $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_{\Delta'}]\!] \sqsubseteq^{\mathsf{Bad}} [\![\mathcal{A}_\Delta]\!]$.*
3. *For any granularity η and $\Delta > 0$ such that $2(\eta + \Delta) < \varepsilon$, we have $[\![\mathcal{A}]\!] \sim_0 [\![\mathsf{impl}_\eta(\mathcal{A})]\!]$ and $[\![\mathsf{impl}_\eta(\mathcal{A})]\!] \approx_\varepsilon [\![\mathsf{impl}_\eta(\mathcal{A})_\Delta]\!]$. Moreover, whenever $\Delta < \frac{1}{|\mathcal{C}|}$, $[\![\mathsf{impl}_\eta(\mathcal{A})_\Delta]\!] \sqsubseteq^{\mathsf{Bad}} [\![\mathcal{A}_\Delta]\!]$.*
4. *For any granularities η and α such that $\eta = k\alpha$ for some $k \in \mathbb{N}_{>0}$ and $\eta < \varepsilon/2$, $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!] \approx_\varepsilon [\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]^\alpha$.*

Note that both $\overline{\mathsf{impl}}_\eta(\mathcal{A})$ and $\mathsf{impl}_\eta(\mathcal{A})$ provide the relation $\approx_{\varepsilon^+}$ between the specification (that is, $[\![\mathcal{A}]\!]$) and the implementation (that is, $[\![\mathcal{A}'_\Delta]\!]$). However, the latter has a stronger relation with $[\![\mathcal{A}]\!]$, so we also study it separately.

Trading precision against complexity. The choice of the granularity in $\overline{\mathsf{impl}}_\eta(\mathcal{A})$ and $\mathsf{impl}_\eta(\mathcal{A})$ allows one to obtain an implementation of $\mathcal{A}$ with any desired precision. However, this comes with a cost since the size of $\overline{\mathsf{impl}}_\eta(\mathcal{A})$ is exponential in the granularity η. But it is also possible to give up on precision in order to reduce the size of the implementation. In fact, one could define $\mathsf{impl}_\equiv(\mathcal{A})$ where the regions are replaced by the equivalence classes of any finite time-abstract bisimulation $\equiv$. Then, we get $[\![\mathcal{A}]\!] \approx_0 [\![\mathsf{impl}_\equiv(\mathcal{A})]\!]$ and $[\![\mathsf{impl}_\equiv(\mathcal{A})]\!]$ is time-abstract bisimilar to $[\![\mathsf{impl}_\equiv(\mathcal{A})_\Delta]\!]$ for any $\Delta > 0$. In order to obtain, say $[\![\mathsf{impl}_\equiv(\mathcal{A})]\!] \approx_K [\![\mathsf{impl}_\equiv(\mathcal{A})_\Delta]\!]$, for some desired $K \geq 1$, one could, roughly, split these bisimulation classes to sets of delay-width at most $O(K)$, that is the maximal delay within a bounded bisimulation class (there is a subtlety with unbounded classes, where, moreover, all states must have arbitrarily large time-successors within the class). Note however that safety specifications are only guaranteed to be preserved for small enough K (see Lemma 1).

5 Proof of Correctness

This section is devoted to the proof of Lemma 1. We start with general properties of regions, in subsection 5.1. In subsection 5.2, we prove the robustness of $\mathsf{impl}_\eta(\mathcal{A})$, $\overline{\mathsf{impl}}_\eta(\mathcal{A})$ and $\mathsf{safe}(\mathcal{A})$, as stated in points 1 through 3 of Lemma 1. In subsection 5.3, we prove that $\overline{\mathsf{impl}}_\eta(\mathcal{A})$ is bisimulation-samplable (point 4). Last, the ready simulation is proved for all the systems in subsection 5.4.

5.1 Properties of regions

We give several properties of the enlargement of regions. Fixing constants η and M, we refer to any (η, M)-region simply as a region.

Proposition 2. *Let $u \in \mathbb{R}^{\mathcal{C}}_{\geq 0}$ such that $u \in [\![\langle \varphi_s \rangle_\Delta]\!]$ for some region s. Then for any subset of clocks $R \subseteq C$, $u[R \leftarrow 0] \in [\![\langle \varphi_{s[R \leftarrow 0]} \rangle_\Delta]\!]$.*

The following proposition shows, intuitively, that enlarged guards cannot distinguish the points of an "enlarged region". The proof is straightforward using difference bound matrices in canonical form. Note that the property does not hold if φ_s is not in canonical form.

Proposition 3. *Let s denote a region, and φ a guard. If $[\![\varphi_s]\!] \subseteq [\![\varphi]\!]$, then $[\![\langle \varphi_s \rangle_\Delta]\!] \subseteq [\![\langle \varphi \rangle_\Delta]\!]$.*

Proposition 4. *Let $u \in \mathbb{R}^{\mathcal{C}}_{\geq 0}$ such that $u \in [\![\langle \varphi_s \rangle_\Delta]\!]$ for some region s. Then for all $s' \in \mathsf{tsucc}^*(s)$, there exists $d \geq 0$ such that $u + d \in [\![\langle \varphi_{s'} \rangle_\Delta]\!]$.*

The previous proposition is no longer valid if φ_s is not canonical. As an example, take the region defined by $x = 1 \wedge y = 0$, whose immediate successor is $1 < x < 2 \wedge 0 < y < 1 \wedge x - y = 1$. The enlargement of the former formula is satisfied by valuation $(x = 1 - \Delta, y = \Delta)$ but this has no time-successor that satisfies the enlargement of the latter.

Last, we need the following proposition which provides a bound on the delay that it takes to go from a region to another.

Proposition 5. *Let r be a region, and s a time-successor region of r, and $\Delta \geq 0$. Suppose that $u \in [\![\varphi_r]\!]$ and $u + d \in [\![\varphi_s]\!]$ for some $d \geq 0$. Then for any $v \in [\![\langle \varphi_r \rangle_\Delta]\!]$, there exists $d' \geq 0$ such that $v + d' \in [\![\langle \varphi_s \rangle_\Delta]\!]$ and $|d' - d| \leq 2\eta + 2\Delta$.*

5.2 Proof of Robustness

We first prove that $\mathsf{impl}_\eta(\mathcal{A})$ and $\overline{\mathsf{impl}}_\eta(\mathcal{A})$ are bisimulation-robust, for an appropriate ε, that is $[\![\mathcal{A}']\!] \approx_\varepsilon [\![\mathcal{A}'_\Delta]\!]$ where $\mathcal{A}'$ denotes any of these (Lemma 2). Then we show "faithfulness" results: Lemma 3 shows that $[\![\mathcal{A}]\!] \sim_0 [\![\mathsf{impl}_\eta(\mathcal{A})]\!]$ and $[\![\mathsf{safe}(\mathcal{A})]\!] \sim_0 [\![\mathcal{A}]\!]$, and Lemma 4 shows that $[\![\mathcal{A}]\!] \approx_{0^+} [\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]$.

Lemma 2. *For any timed automaton $\mathcal{A}$, any granularity η, and any $\Delta > 0$, we have $[\![\mathsf{impl}_\eta(\mathcal{A})]\!] \approx_{2\Delta+2\eta} [\![\mathsf{impl}_\eta(\mathcal{A})_\Delta]\!]$ and $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!] \approx_{2\Delta+2\eta} [\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_\Delta]\!]$.*

Proof (Sketch). We fix any η and Δ. Let us consider $\mathsf{impl}_\eta(\mathcal{A})$. The case of $\overline{\mathsf{impl}}_\eta(\mathcal{A})$ is similar. We define relation $\mathcal{R} \subseteq (\mathcal{L} \times \mathbb{R}^C) \times (\mathcal{L} \times \mathbb{R}^C)$ between $[\![\mathsf{impl}_\eta(\mathcal{A})]\!]$ and $[\![\mathsf{impl}_\eta(\mathcal{A})_\Delta]\!]$ by $(l_r, u)\mathcal{R}(l'_{r'}, u')$ whenever $l_r = l'_{r'}$ and

$$\forall s \in \mathsf{tsucc}^*(r), \exists d \geq 0, u + d \in [\![\varphi_s]\!] \iff \exists d' \geq 0, u' + d' \in [\![\langle \varphi_s \rangle_\Delta]\!]. \quad (1)$$

Intuitively, relation $\mathcal{R}$ relates states which can reach, by a delay, the same set of regions: we require the first system to reach $[\![\varphi_s]\!]$, while it is sufficient that the

second one reaches $[\![\langle\varphi_s\rangle_\Delta]\!]$, since its guards are enlarged by Δ. Then, Propositions 2, 3, and 4 ensure that this relation is maintained after each transition, proving that $\mathcal{R}$ is a timed-action bisimulation. The parameter $2\Delta + 2\eta$ is given by Proposition 5, applied on relation (1). □

The parameter which we provide for the timed-action bisimilarity is (almost) tight. In fact, consider the automaton in Figure 2, where the guard of the edge entering ℓ_1 is changed to $x \leq 1$. Fix any η and Δ and consider the following cycle in $\overline{\mathsf{impl}}_\eta(\mathcal{A})$: $(\ell_{1,r_1}) \to (\ell_{2,r_2}) \to (\ell_{1,r_1})$, where r_1 is the region $1-\eta < x < 1 \wedge y = 0$, and r_2 is the region $x = 0 \wedge 1 < y < 1+\eta$. Suppose $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_\Delta]\!]$ first goes to location (ℓ_{1,r_1}) with $x = 1+\Delta, y = 0$, and that this is matched in $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]$ by $(\ell_{1,r'_1}, (x = 1-\alpha, y = 0))$ where necessarily $\alpha \geq 0$. It is shown in [11] that in any such cycle, the enlarged automaton can reach (by iterating the cycle) all states of the region r_1 at location ℓ_1. In particular, $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_\Delta]\!]$ can go to state $(\ell_{1,r_1}, (x = 1-\eta, y = 0))$. However, without enlargement, all states $(\ell_1, r'_1, (v'_x, v'_y))$ reached from a state $(\ell_{1,r_1}, (v_x, v_y))$ with $v_y = 0$ satisfy $v'_x \geq v_x$, that is, the value of the clock x at location ℓ_1 cannot decrease along any run ([11]). Thus, the state $(\ell_{1,r_1}, (x = 1-\eta, y = 0))$ of $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_\Delta]\!]$ is matched in $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]$ by some state $(\ell_{1,r''_1}, (v'_x, 0))$ where $v'_x \geq 1-\alpha$. Now, from there, $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_\Delta]\!]$ can delay $1+\Delta+\eta$ and go to ℓ_2, whereas $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]$ can delay at most $1+\alpha$ to take the same transition. The difference between the delays at the first and the last step is then at least $\max\big(\Delta+\alpha, 1+\Delta+\eta-(1+\alpha)\big) \geq \Delta+\eta/2$.

Next, we show that $\mathsf{safe}(\mathcal{A})$ and $\mathsf{impl}_\eta(\mathcal{A})$ are strongly 0-bisimilar to $\mathcal{A}$. The proof is omitted.

Lemma 3. *For any timed automaton $\mathcal{A}$, we have $[\![\mathsf{safe}(\mathcal{A})]\!] \sim_0 [\![\mathcal{A}]\!]$, and $[\![\mathcal{A}]\!] \sim_0 [\![\mathsf{impl}_\eta(\mathcal{A})]\!]$ for any granularity η.* □

The proof of $[\![\mathcal{A}]\!] \approx_{0^+} [\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]$ is trickier. In fact, since all guards are closed in $\overline{\mathsf{impl}}_\eta(\mathcal{A})$, but not necessarily in $\mathcal{A}$, all time delays may not be matched exactly. The first part of the proof follows the lines of Proposition 16 of [19], who, by a similar construction, prove that the finite timed traces of $[\![\mathcal{A}]\!]$ are dense in those of $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]$, for an appropriate topology. Their result has a similar flavor, but we consider 0^+-bisimulation which cannot be interpreted in terms of density in an obvious way.

Lemma 4. *For any timed automaton $\mathcal{A}$ and granularity η, $[\![\mathcal{A}]\!] \approx_{0^+} [\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]$.*

Proof (Sketch). We fix any η and $\delta \in (0,1)$. We define $(l,v)\mathcal{R}(l_r, v')$ iff

$$r = \mathsf{reg}(v), v' \in \overline{\mathsf{reg}(v)} \text{ and } \exists v'' \in \mathsf{reg}(v), v = \delta v'' + (1-\delta)v'. \tag{2}$$

We show that $\mathcal{R}$ is a timed-action 0^+-bisimulation. One direction of the bisimulation follows from convexity of regions, while the other direction is less obvious, and necessitates the following technical lemma. □

Lemma 5. *Let $v, v', v'' \in \mathbb{R}_{\geq 0}$ such that $v'' \in \mathsf{reg}(v)$ and $v' \in \overline{\mathsf{reg}(v)}$, and $v = \varepsilon v'' + (1-\varepsilon)v'$ for some $\varepsilon \in (0,1)$. Then for all $d \geq 0$, there exists $d', d'' \geq 0$ s.t. $v+d = \varepsilon(v''+d'') + (1-\varepsilon)(v'+d')$, $v''+d'' \in \mathsf{reg}(v+d)$ and $v'+d' \in \overline{\mathsf{reg}(v+d)}$.*

5.3 Proof of Samplability

We now show that $\overline{\mathsf{impl}}_\eta(\mathcal{A})$ is a sampled implementation for any timed automaton $\mathcal{A}$. This result follows from the following lemma and Lemma 4. The proof is similar to Lemma 2.

Lemma 6. *Let $\mathcal{A}$ be any integral timed automaton. For any granularities η and α such that $\eta = k\alpha$ for some $k \in \mathbb{N}_{>0}$, we have $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!] \approx_{2\eta} [\![\overline{\mathsf{impl}}_\eta(\mathcal{A})]\!]^\alpha$.*

5.4 Proof of Safety Preservation (Ready Simulation)

Lemma 7. *We have $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_{\Delta'}]\!] \sqsubseteq^{\mathsf{Bad}} [\![\mathcal{A}_\Delta]\!]$ and $[\![\mathsf{impl}_\eta(\mathcal{A})_\Delta]\!] \sqsubseteq^{\mathsf{Bad}} [\![\mathcal{A}_\Delta]\!]$ for any $0 < \Delta' < \Delta < \frac{1}{|\mathcal{C}|}$; and $[\![\mathsf{safe}(\mathcal{A})_\Delta]\!] \sqsubseteq^{\mathsf{Bad}} [\![\mathcal{A}_\Delta]\!]$ for any $\Delta < \frac{1}{2|\mathcal{C}|}$.*

Proof (Sketch). The simulation can be shown similarly to Lemma 3. We show that actions Bad are not enabled in any state of the simulating run, whenever $\mathcal{A}$ is safe w.r.t. Bad. Let us consider the first statement. Informally, this is due to two facts: (1) the set of reachable states in $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_\Delta]\!]$ have a small distance (at most Δ) to the corresponding reachable states in $[\![\mathcal{A}]\!]$; (2) the states of $[\![\mathcal{A}]\!]$ have a positive distance to $\mathrm{Pre}_{\mathcal{A}}(\mathsf{Bad})$, which can be bounded from below by $\frac{1}{|\mathcal{C}|}$. Thus, choosing $\frac{1}{|\mathcal{C}|} - \Delta > 0$, we prove that $[\![\overline{\mathsf{impl}}_\eta(\mathcal{A})_{\Delta'}]\!]$ is also safe w.r.t. Bad. □

6 Conclusion

We have presented a way to transform any timed automaton into robust and samplable ones, while preserving the original semantics with any desired precision. Such a transformation is interesting if the timed automaton under study is not robust (or not samplable), or cannot be certified as such. In this case, one can simply model-check the original automaton for desired properties, then apply our constructions, which will preserve the specification.

Our constructions also allow one to solve the *robust synthesis* problem. In the synthesis problem, the goal is to obtain automatically (i.e. to synthesize) a timed automaton which satisfies a given property. If one solves this problem for timed automata and obtain a synthesized system $\mathcal{A}$, then applying our constructions, we get that $\overline{\mathsf{impl}}_\eta(\mathcal{A})_\Delta$ and $\overline{\mathsf{impl}}_\eta(\mathcal{A})^\eta$ satisfy the same (say, untimed) properties.

As a future work, we will be interested in *robust controller synthesis*. In this problem, we are given a system $\mathcal{S}$ which we cannot change, and we are asked to synthesize a system $\mathcal{C}$, called controller, such that the parallel composition of the two satisfies a given property. The *robust* controller synthesis is the controller synthesis problem where the behaviour of the controller is $\mathcal{C}_\Delta$ (the controller has imprecise clocks), and we need to decide whether there is some Δ for which the parallel composition still satisfies the property.

Acknowledgement. We thank David N. Jansen for his detailed and insightful comments.

References

1. P. Abdulla, P. Krčál, and W. Yi. Sampled semantics of timed automata. *Logical Methods in Computer Science*, 6(3:14), 2010.
2. K. Altisen and S. Tripakis. Implementation of timed automata: An issue of semantics or modeling? In FORMATS'05, LNCS 3829, p. 273–288. Springer, 2005.
3. R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.
4. E. Asarin, O. Maler, and A. Pnueli. On discretization of delays in timed automata and digital circuits. In CONCUR'98, LNCS 1466, p. 470–484. Springer, 1998.
5. C. Baier, N. Bertrand, P. Bouyer, Th. Brihaye, and M. Größer. Probabilistic and topological semantics for timed automata. In FSTTCS'07, LNCS 4855, p. 179–191. Springer, 2007.
6. P. Bouyer, N. Markey, and P.-A. Reynier. Robust model-checking of linear-time properties in timed automata. In LATIN'06, LNCS 3887, p. 238–249. Springer, 2006.
7. P. Bouyer, N. Markey, and P.-A. Reynier. Robust analysis of timed automata via channel machines. In FoSSaCS'08, LNCS 4962, p. 157–171. Springer, 2008.
8. M. Bozga, O. Maler, A. Pnueli, and S. Yovine. Some progress in the symbolic verification of timed automata. In CAV'97, LNCS 1254, p. 179–190. Springer, 1997.
9. F. Cassez, T. A. Henzinger, and J.-F. Raskin. A comparison of control problems for timed and hybrid systems. In HSCC'02, LNCS 2289, p. 134–148. Springer, 2002.
10. L. de Alfaro, T. A. Henzinger, and R. Majumdar. Discounting the future in systems theory. In ICALP'03, LNCS, LNCS, p. 1022–1037. Springer, 2003.
11. M. De Wulf, L. Doyen, N. Markey, and J.-F. Raskin. Robust safety of timed automata. *Formal Methods in System Design*, 33(1-3):45–84, 2008.
12. M. De Wulf, L. Doyen, and J.-F. Raskin. Almost ASAP semantics: From timed models to timed implementations. *Formal Aspects of Computing*, 17(3):319–341, 2005.
13. D. L. Dill. Timing assumptions and verification of finite-state concurrent systems. In AVMFSS'89, LNCS 407, p. 197–212. Springer, 1990.
14. U. Fahrenberg, K. G. Larsen, and C. Thrane. A quantitative characterization of weighted Kripke structures in temporal logic. *Journal of Computing and Informatics*, 29(6+):1311–1324, 2010.
15. V. Gupta, Th. A. Henzinger, and R. Jagadeesan. Robust timed automata. In HART'97, LNCS 1201, p. 331–345. Springer, 1997.
16. T. A. Henzinger, R. Majumdar, and V. Prabhu. Quantifying similarities between timed systems. In FORMATS'05, LNCS 3829, p. 226–241. Springer, 2005.
17. P. Krčál and R. Pelánek. On sampled semantics of timed systems. In FSTTCS'05, LNCS 3821, p. 310–321. Springer, 2005.
18. K. G. Larsen and A. Skou. Bisimulation through probabilistic testing. In POPL'89, p. 344–352, 1989.
19. J. Ouaknine and J. Worrell. Revisiting digitization, robustness, and decidability for timed automata. In LICS'03, p. 198–207. IEEE Computer Society, 2003.
20. A. Puri. Dynamical properties of timed systems. *Discrete Event Dynamic Systems*, 10(1-2):87–113, 2000.
21. O. Sankur, P. Bouyer, and N. Markey. Shrinking timed automata. Submitted, 2011.
22. C. Thrane, U. Fahrenberg, and K. G. Larsen. Quantitative analysis of weighted transition systems. *Journal Logic and Algebraic Programming*, 79(7):689–703, 2010.