

Sampled Universality of Timed Automata

Parosh Aziz Abdulla, Pavel Krcal, and Wang Yi

Uppsala University, Sweden.
Email: {parosh,pavelk,yi}@it.uu.se.

Abstract. Timed automata can be studied in not only a dense-time setting but also a discrete-time setting. The most common example of discrete-time semantics is the so called *sampled* semantics (i.e., discrete semantics with a fixed time granularity ε). In the real-time setting, the universality problem is known to be undecidable for timed automata. In this work, we study the universality question for the languages accepted by timed automata with sampled semantics. On the negative side, we show that deciding whether for all sampling periods ε a timed automaton accepts all timed words in ε -sampled semantics is as hard as in the real-time case, i.e., undecidable. On the positive side, we show that checking whether there is a sampling period such that a timed automaton accepts all untimed words in ε -sampled semantics is decidable. Our proof uses clock difference relations, developed to characterize the reachability relation for timed automata in connection with sampled semantics.

1 Introduction

Timed automata [3] are considered as one of the standard models for timed systems. The semantics of these models can be defined over various time domains. The most common one is the set of nonnegative real numbers, giving *dense* time semantics. The dense time semantics allows for the description of how a system behaves at every real-valued time point with arbitrarily fine precision, and thus one needs not consider time granularity in modeling and verification. To study systems which have a fixed granularity of time (e.g., clock cycles), discrete time semantics, and in particular, *sampled* semantics with fixed time step ε are often considered, e.g., [10, 5]. In such a case, the time domain is $\{k \cdot \varepsilon \mid k \in \mathbb{N}_0\}$, where $\varepsilon = 1/n$ for some $n \in \mathbb{N}$.

In this paper, we study the universality question for the languages accepted by timed automata in sampled semantics. Let A be a timed automaton and $L_\varepsilon(A)$ denote the sampled language accepted by A in the ε -sampled semantics, i.e. the set of timed traces where all events are associated with a timestamp which is $n \cdot \varepsilon$ for some natural number n . More precisely we study the following problems:

1. *Existential timed universality* which is to check whether $L_\varepsilon(A)$ is universal for some sampling period ε .
2. *Universal timed universality* which is to check whether $L_\varepsilon(A)$ is universal for all sampling periods ε .

3. *Existential untimed universality* which is to check whether the untimed language of $L_\varepsilon(\mathcal{A})$ is universal for some sampling period ε .
4. *Universal untimed universality* which is to check whether the untimed language of $L_\varepsilon(\mathcal{A})$ is universal for all sampling periods ε .

We show that problem 1 and 4 are easy to check. In fact, they are equivalent to checking whether $L_1(\mathcal{A})$ is timed (and untimed) universal (in the sampled semantics with sampling period 1). As our main results, we prove the (un)decidability of the remaining two questions for both finite and infinite words. On the negative side, we show that problem 2 is as hard as the universality problem of timed automata in the real-time setting, that is undecidable (in fact, Π_1^1 -hard). On the positive side, we show that problem 3 is decidable.

The decidability proof extends the standard subset construction technique by a novel procedure for detection of loops which enforce nonimplementable behaviors, i.e., behaviors which cannot be realized by a system whenever we fix a sampling period. This procedure is based on a recently described technique – *clock difference relations* [12]. This technique has been developed in connection with sampled semantics to characterize the reachability relation for timed automata. It can be used to detect behaviors of timed automata in the dense time semantics which are not present in a sampled semantics for any sampling period.

Related Work Universality of timed automata has been shown Π_1^1 -hard in the seminal paper [3] for the real time semantics. Later papers study the universality (or the language inclusion) problem for subclasses of timed automata, e.g., closed/open timed automata [16], robust automata [11], or timed automata with one clock [17, 1]. There has been a considerable amount of work related to discretization issues and verifying dense time properties using discrete time methods, e.g., [10, 13, 15, 5]. The main difference compared to our work is that usually only a fixed sampling rate is considered. Practical aspects of verification with the use of sampled semantics are discussed in [7, 6, 4]. These works are concerned mainly with data structures for representing sets of discrete valuations (e.g., different types of decision diagrams). Implementability issues are discussed in connection with robust semantics of timed automata in [20, 18]. The reachability relations for timed automata were also characterized using the additive theory of real numbers in [8] and using $2n$ -automata in [9].

2 Preliminaries

We consider the standard model of timed automata [3].

Definition 1. A timed automaton \mathcal{A} is a tuple $\langle \mathcal{C}, \Sigma, N, l_0, E, F \rangle$ where

- \mathcal{C} is a set of real-valued clocks,
- Σ is a finite alphabet of events,
- N is a finite set of locations,
- $l_0 \in N$ is the initial location,

- $E \subseteq N \times \Phi(\mathcal{C}) \times \Sigma \times 2^{\mathcal{C}} \times N$ is the set of edges (describing possible transitions),
and
- $F \subseteq N$ is the set of accepting locations.

The set $\Phi(\mathcal{C})$ of clock constraints (guards) g is defined as a set of conjunctive formulas of atomic constraints in the form $x \sim m$ where $x \in \mathcal{C}$ is a clock, $\sim \in \{\leq, <, \geq, >\}$, and m is a natural number. A clock valuation $\nu \in [\mathcal{C} \rightarrow \mathbb{R}_{\geq 0}]$ is a function mapping clocks to non-negative real numbers. We use $\nu + t$ to denote the clock valuation which maps each clock x to the value $\nu(x) + t$, and $\nu[r \mapsto 0]$ for $r \subseteq \mathcal{C}$ to denote the clock valuation which maps each clock in r to 0 and agrees with ν for the other clocks (i.e., $\mathcal{C} \setminus r$). An edge (l_1, g, e, r, l_2) represents a transition from location $l_1 \in N$ to location $l_2 \in N$ accepting an input symbol (an *event*) $e \in \Sigma$, and resetting clocks in $r \subseteq \mathcal{C}$ to zero. The transition can be performed only if the current values of clocks satisfy g .

We present the definition of runs, accepting runs, and the language of timed automaton relative to a time domain on which the automaton operates. A *time domain* \mathcal{T} is a subset of nonnegative real numbers satisfying the following properties. If $a, b \in \mathcal{T}$ then $a + b \in \mathcal{T}$ and $0 \in \mathcal{T}$. If we take $\mathcal{T} = \mathbb{R}_{\geq 0}$ then we get the standard semantics of timed automata. A \mathcal{T} -*timed event* is a pair (t, e) , where $e \in \Sigma$ is an event and $t \in \mathcal{T}$ is called a *timestamp* of the event e . A \mathcal{T} -*timed trace* is a (possibly infinite) sequence of \mathcal{T} -timed events $\xi = (t_1, e_1)(t_2, e_2)\dots$, $e_i \in \Sigma$, $t_i \in \mathcal{T}$ and $t_i \leq t_{i+1}$ for all $i \geq 1$.

Definition 2. A run of a timed automaton $\mathcal{A} = \langle \mathcal{C}, \Sigma, N, l_0, E, F \rangle$ over a \mathcal{T} -timed trace $\xi = (t_1, e_1)(t_2, e_2)(t_3, e_3)\dots$, is a (possibly infinite) sequence of the form

$$(l_0, \nu_0) \xrightarrow[t_1]{e_1} (l_1, \nu_1) \xrightarrow[t_2]{e_2} (l_2, \nu_2) \xrightarrow[t_3]{e_3} \dots$$

where (l_i, ν_i) are states of \mathcal{A} , $l_i \in N$, ν_i is a clock valuation, satisfying the following conditions:

- $\nu_0(x) = 0$ for all $x \in \mathcal{C}$.
- for all $i \geq 1$, there is an edge $(l_{i-1}, g_i, e_i, r_i, l_i)$ such that $(\nu_{i-1} + t_i - t_{i-1})$ satisfies g_i (we define $t_0 = 0$) and $\nu_i = (\nu_{i-1} + t_i - t_{i-1})[r_i \mapsto 0]$.

A finite run of a timed automaton is accepting if $l_n \in F$ for its last state (l_n, ν_n) . An infinite run of a timed automaton is accepting if an accepting location $l \in F$ occurs on it infinitely often (standard Büchi acceptance condition). The (finite word) timed language of a timed automaton \mathcal{A} with respect to the time domain \mathcal{T} , denoted $L_{\mathcal{T}}(\mathcal{A})$, is the set of all finite \mathcal{T} -time traces for which there is an accepting run of \mathcal{A} . Analogously, the timed ω -language of a timed automaton \mathcal{A} with respect to the time domain \mathcal{T} denoted $L_{\mathcal{T}}^{\omega}(\mathcal{A})$ is the set of all infinite \mathcal{T} -time traces for which there is an accepting run of \mathcal{A} .

An untimed function \mathcal{U} maps a timed trace into a word over Σ by projecting out the timestamps, i.e., $\mathcal{U}((t_1, e_1)(t_2, e_2)(t_3, e_3)\dots) = e_1 e_2 e_3 \dots$. A natural extension of \mathcal{U} to sets of timed traces maps timed languages into their untimed counterparts.

We say that languages $L_{\mathcal{T}}(\mathcal{A})$, $L_{\mathcal{T}}^{\omega}(\mathcal{A})$ are *timed universal* if they contain all finite, resp. infinite, \mathcal{T} -timed traces over Σ and \mathcal{T} . The languages $L_{\mathcal{T}}(\mathcal{A})$, $L_{\mathcal{T}}^{\omega}(\mathcal{A})$ are *untimed universal* if $\mathcal{U}(L_{\mathcal{T}}(\mathcal{A})) = \Sigma^*$, $\mathcal{U}(L_{\mathcal{T}}^{\omega}(\mathcal{A})) = \Sigma^{\omega}$, respectively.

The time domains of special interest in this paper are the sampled (digital) time domains $\mathcal{T}_{\varepsilon} = \{k \cdot \varepsilon | k \in \mathbb{N}_0\}$, where ε is a sampling period, $\varepsilon = 1/n$ for some $n \in \mathbb{N}$. To simplify the notation, we write $L_{\varepsilon}(\mathcal{A})$, $L_{\varepsilon}^{\omega}(\mathcal{A})$ instead of $L_{\mathcal{T}_{\varepsilon}}(\mathcal{A})$, $L_{\mathcal{T}_{\varepsilon}}^{\omega}(\mathcal{A})$, respectively. In the following, we always assume that $\varepsilon = 1/n$ for some $n \in \mathbb{N}$. We will also write $L(\mathcal{A})$, $L^{\omega}(\mathcal{A})$ when $\mathcal{T} = \mathbb{R}_{\geq 0}$ (standard real time semantics). Figure 1 shows an automaton \mathcal{A} such that $L(\mathcal{A})$ is untimed universal, but there is no ε such that $L_{\varepsilon}(\mathcal{A})$ is also untimed universal. For any ε , there is $k \in \mathbb{N}$ such that the word a^{2k} does not belong to $\mathcal{U}(L_{\varepsilon}(\mathcal{A}))$.

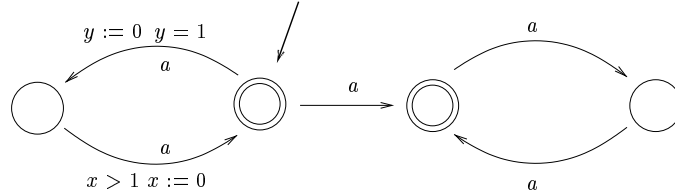


Fig. 1. An example of an automaton which is untimed universal in the real time semantics, but there is no ε such that it is untimed universal in the ε -sampled semantics.

For a timed automaton $\mathcal{A} = \langle \mathcal{C}, \Sigma, N, l_0, E, F \rangle$ we will use the standard notion of the region automaton [3]. States of a region automaton consist of a location and a region. A region is a set of valuations (an equivalence class of so called region equivalence). These sets can be represented in a finite way, because the region equivalence has finite index for each timed automaton. Regions will be denoted by D, D', \dots . Transitions in a region automaton $(l, D) \xrightarrow{a} (l', D')$ or $(l, D) \xrightarrow{t} (l', D')$ are labeled by either an $a \in \Sigma$ or by a special symbol t denoting an immediate time successor (a time delay). In the following, we denote states of the region automaton by R, R_1, R_2, \dots and the initial state of the region automaton by R_0 . By a path π in the region automaton starting from R and leading to R' labeled by w (denoted $\pi = R \xrightarrow{w} R'$) we mean a sequence of transitions starting from R and leading to R' labeled by w' such that $w = w' \upharpoonright t$ (w is equal to w' with all labels t projected out) and the last letter of w' is different from t (the last transition of the path is labeled by some $a \in \Sigma$). Note that a path does not have to be uniquely determined by R, R' and w .

When we say that there is a run of a timed automaton \mathcal{A} over a path $\pi = R \xrightarrow{w} R'$ in an ε -sampled semantics (real time semantics) then we mean that there is a run of \mathcal{A} over w' where w' is a $\mathcal{T}_{\varepsilon}$ -timed trace ($\mathbb{R}_{\geq 0}$ -timed trace) such that $w = \mathcal{U}(w')$ going through π . If we need to specify the starting and the finishing state, then we write $(l, \nu) \xrightarrow{\pi}_{\varepsilon} (l', \nu')$ or $(l, \nu) \xrightarrow{\pi} (l', \nu')$.

3 Reachability Relations

As a technical tool in our proofs, we will use clock difference relations [12] to characterize reachability relations. The notion of the reachability relation describes exactly the valuations which can be reached in a timed automaton \mathcal{A} from a given valuation while going through a given path in the region automaton for \mathcal{A} . Let $\pi = (l, D) \xrightarrow{w} (l', D')$ be a path in the region automaton for \mathcal{A} . We want to characterize which concrete states $(l', \nu'), \nu' \in D'$ can be reached from a concrete state $(l, \nu), \nu \in D$ by a run of \mathcal{A} over π in the real time semantics. We are not interested in the clock values when they grow over the greatest constant in \mathcal{A} , denoted by K . To abstract from such clocks, we define a relation \sim_K on clock valuations as follows: $\nu \sim_K \nu'$ iff for all $x \in \mathcal{C}$: $\nu(x) = \nu'(x)$ or $\nu(x) > K \wedge \nu'(x) > K$. Formally, the *reachability relation* of a path $\pi = (l, D) \xrightarrow{w} (l', D')$ is a relation on valuations $C^\pi \subseteq D \times D'$ such that for each $\nu \in D, \nu' \in D'$:

$$(\nu, \nu') \in C^\pi \iff \exists \nu'' \sim_K \nu' : (l, \nu) \xrightarrow{\pi} (l', \nu'').$$

This relation can be characterized by a structure over a set of clocks \mathcal{C} called *clock difference relations*. This structure is a set of (in)equalities of the following form:

- $x' - y' \bowtie u - v$
- $x' - y' \bowtie 1 - (u - v)$

where $\bowtie \in \{<, >, =\}$, $x, y, u, v \in \mathcal{C}$. We use primed clock names on the left hand side of the (in)equalities to denote the fact that these clocks are interpreted in the target valuation, whereas the unprimed clocks are interpreted in the starting valuation. The semantics of a CDR B is defined as follows. For any $\delta \in \mathbb{R}$, $\text{fr}(\delta)$ denotes the fractional part of δ . We say that a pair of valuations (ν, ν') satisfies B if and only if:

- if $x' - y' \bowtie u - v \in B$ then $\text{fr}(\nu'(x)) - \text{fr}(\nu'(y)) \bowtie \text{fr}(\nu(u)) - \text{fr}(\nu(v))$,
- if $x' - y' \bowtie 1 - (u - v) \in B$ then $\text{fr}(\nu'(x)) - \text{fr}(\nu'(y)) \bowtie 1 - (\text{fr}(\nu(u)) - \text{fr}(\nu(v)))$.

The semantics of clock difference relations can be extended to sets of clock difference relations. A pair of valuations (ν, ν') satisfies a set of clock difference relations if it satisfies at least one of them. By $\sigma = R \xrightarrow{w} R'$ we denote a set of (not necessarily all) paths which start in the same state of the region automaton R , lead to the same state of the region automaton R' , and are labeled by the same word w .

Lemma 1 ([12]). *For a given set of paths $\sigma = R \xrightarrow{w} R'$, the reachability relation $\bigcup_{\pi \in \sigma} C^\pi$ is effectively definable as a (finite) set of clock difference relations.*

It follows from this lemma that for any two given sets of paths $\sigma = R \xrightarrow{w} R', \sigma' = R \xrightarrow{w'} R'$ can one algorithmically check whether $\bigcup_{\pi \in \sigma} C^\pi = \bigcup_{\pi \in \sigma'} C^\pi$ and whether there is ν such that $(\nu, \nu) \in \bigcup_{\pi \in \sigma} C^\pi$.

Now we define two concepts characterizing a property of loops in timed automata and state that they are equivalent for the sampled semantics. Let us assume that for a given timed automaton \mathcal{A} , σ is a set of (not necessarily all) paths from R back to itself labeled by w in the region automaton (\mathcal{A} is non-deterministic, there can be several such paths). We call this sequence a *loop* $\sigma = R \xrightarrow{w} R$. This loop represents (fragments of) runs of \mathcal{A} which start in a state $(l, \nu) \in R$ and end in some state $(l, \nu') \in R$ over some $\pi \in \sigma$. A crucial fact for our decision procedure is whether we can start again from (l, ν') and run over some $\pi \in \sigma$, ending in some $(l, \nu'') \in R$, and whether we can iterate the loop like this unboundedly many times.

Definition 3. *For a timed automaton \mathcal{A} and a loop $\sigma = R \xrightarrow{w} R$, $R = (l, D)$, we say that σ can be iterated in the ε -sampled semantics if for any $k \in \mathbb{N}$ there is a concrete run $(l, \nu_0) \xrightarrow{\pi_1} (l, \nu_1) \xrightarrow{\pi_2} \dots \xrightarrow{\pi_k} (l, \nu_k)$, where $\pi_i \in \sigma, \nu_i \in D$ for all $0 \leq i \leq k$.*

In the real time semantics, any loop $\sigma = R \xrightarrow{w} R$ can be iterated, because for any $(l, \nu) \in R$ there is a state $(l, \nu') \in R$ such that there is a run of \mathcal{A} over any $\pi \in \sigma$ starting from (l, ν) and ending in (l, ν') . Therefore, one can compose these runs into an arbitrarily long one. This is not true in the sampled time domains. There are timed automata such that some loops in their region automata can be for any ε iterated only finitely many times. Intuitively, the sufficient and necessary condition is that the automaton can get back exactly to the same concrete valuation in which it started after several iterations of the loop. This can be characterized in the terms of the reachability relations.

Definition 4. *For a timed automaton \mathcal{A} and a loop $\sigma = R \xrightarrow{w} R$, $R = (l, D)$, we say that σ is exact if and only if there is $k \in \mathbb{N}$ and a clock valuation $\nu \in D$ such that $(\nu, \nu) \in (\bigcup_{\pi \in \sigma} C^\pi)^k$.*

The following lemma characterizes precisely when loops in a region automaton can be iterated unboundedly many times also in the sampled time domains.

Lemma 2 ([12]).

For a timed automaton \mathcal{A} and a loop $\sigma = R \xrightarrow{w} R$, where $R = (l, D)$, there is an ε such that σ can be iterated in ε -sampled semantics if and only if σ is exact.

The following observation can give some intuition for the fact that $\sigma = R \xrightarrow{w} R$ cannot be iterated when there is no k such that $(\nu, \nu) \notin (\bigcup_{\pi \in \sigma} C^\pi)^k$. For any ε , there are only finitely many different valuations of the clocks in R . But if for all $k \in \mathbb{N}$, $(\nu, \nu) \notin (\bigcup_{\pi \in \sigma} C^\pi)^k$ then each iteration of $R \xrightarrow{w} R$ has to result in a new clock valuation. Thus, it is not possible to iterate the loop again after sufficiently many iterations. After every iteration of the left loop of the automaton in Figure 1, the difference between the fractional parts of the clocks x and y grows but it has to be smaller than 1 all the time. Therefore, the computation will be blocked after at most $1/\varepsilon$ iterations.

4 Existential Sampled Universality for Finite Words

In this part we study the decidability of the question whether there exists an ε such that the untimed ε -sampled language is universal. We give a positive answer for finite word languages in this section and for infinite word languages in the next section.

Theorem 1. *For a given timed automaton \mathcal{A} , the question whether there exists an ε such that $L_\varepsilon(\mathcal{A})$ is untimed universal is decidable.*

Note that the untimed universality is decidable for dense time semantics (both $L(\mathcal{A})$ and $L^\omega(\mathcal{A})$). For some timed automata, sampled semantics cuts out some words from their untimed language, i.e., it can happen that for each ε there is some untimed word w such that $w \in \mathcal{U}(L(\mathcal{A}))$ and $w \notin \mathcal{U}(L_\varepsilon(\mathcal{A}))$.

To solve the problem, we present an algorithm such that for a given timed automaton \mathcal{A} the algorithm answers 'YES' if there is an ε such that $L_\varepsilon(\mathcal{A})$ is untimed universal. Otherwise, it answers 'NO' and gives a simple description of a set of words which for each ε contains a counterexample of the untimed universality of $L_\varepsilon(\mathcal{A})$.

Before we present the algorithm, we define an auxiliary concept capturing the nature of the loops in the subset construction for timed automata. We assume a fixed timed automaton \mathcal{A} and its region automaton. Let σ be a set of (not necessarily all) paths in the region automaton labeled by w . Then for all w' prefixes of w , $\mathcal{R}(\sigma, w')$ means a set of states of the region automaton reachable by the prefix π' of a path $\pi \in \sigma$ such that π' is labeled by w' . We say that a set of paths σ over w is a *sequence* if for all paths π over w in the region automaton the following holds: if $\mathcal{R}(\sigma, w') = \mathcal{R}(\sigma \cup \{\pi\})$ for all w' prefixes of w then $\pi \in \sigma$ (we say that a sequence is *state complete*). A state complete simple sequences $\sigma = R \xrightarrow{w} R'$ is a special case of sequences where all paths start in the same state R and end in the same state R' . Let us assume that σ is a sequence over w and u, v are words such that uv is a prefix of w . For two states of the region automaton $R \in \mathcal{R}(\sigma, u)$, $R' \in \mathcal{R}(\sigma, uv)$ we define a simple sequence $\sigma' = R \xrightarrow{v} R'$ *relative to σ, u, v, R, R'* as a set of paths π from R to R' labeled by v such that there are paths π' and π'' , π' is labeled by u , and $\pi'\pi'' \in \sigma$. Finally, let $\text{Pattern}(\sigma, u, v)$ denote a partial mapping which for a pair of states of the region automaton returns a reachability relation according to the following rule:

$$\text{Pattern}(\sigma, u, v)(R, R') = \begin{cases} \bigcup_{\pi \in \sigma'} C^\pi & \text{if } R \in \mathcal{R}(\sigma, u) \text{ and } R' \in \mathcal{R}(\sigma, uv); \\ & \sigma' = R \xrightarrow{v} R' \text{ is a simple sequence} \\ & \text{relative to } \sigma, u, v, R, R' \\ \perp & \text{otherwise} \end{cases}$$

Intuitively, $\text{Pattern}(\sigma, u, v)$ summarizes the reachability pattern relative to σ (all paths involved have to be the corresponding fragments of the paths in σ)

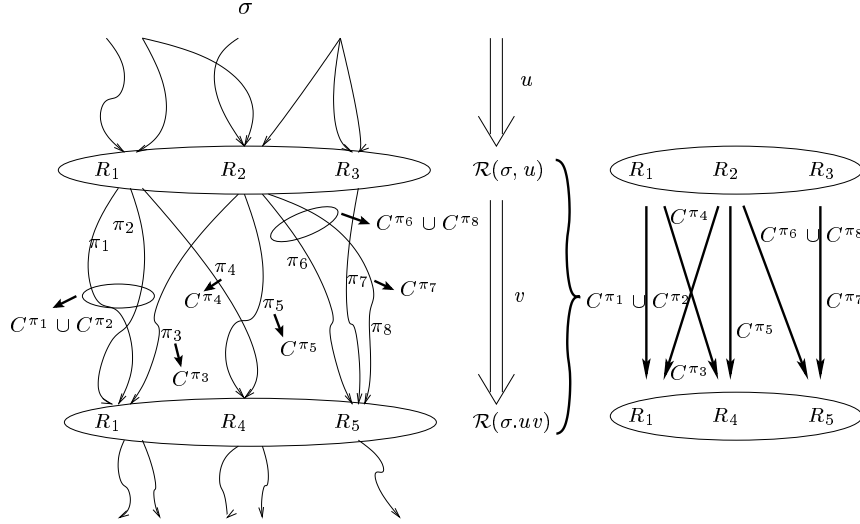


Fig. 2. A schema of $\text{Pattern}(\sigma, u, v)$. For instance, $\text{Pattern}(\sigma, u, v)(R_2, R_5) = C^{\pi_6} \cup C^{\pi_8}$. When there is no path in σ between two states of the region automaton labeled by v (e.g., between R_3 and R_4) then the corresponding reachability relation is empty ($\text{Pattern}(\sigma, u, v)(R_3, R_4) = \emptyset$).

between states reachable by u and states reachable by uv . A schema of such a mapping is depicted in Figure 2.

The algorithm works with the set of the sequences (denote Visited). Note that for a given region automaton each sequence σ over w is fully determined by the sets $\mathcal{R}(\sigma, w')$ for all w' prefixes of w . Also, all operations over sequences result in manipulations with these sets. We define one concept and two simple operations which the algorithm uses. For a sequence σ over w :

- a division of w into $w = w_1 v_1 v_2 v_3 w_2$ is a *full loop* of σ if $|v_1| > 0, |v_2| > 0, |v_3| > 0, \mathcal{R}(\sigma, w_1) = \mathcal{R}(\sigma, w_1 v_1) = \mathcal{R}(\sigma, w_1 v_1 v_2) = \mathcal{R}(\sigma, w_1 v_1 v_2 v_3)$, $\text{Pattern}(\sigma, w_1, v_1) = \text{Pattern}(\sigma, w_1 v_1, v_2) = \text{Pattern}(\sigma, w_1 v_1 v_2, v_3) = \text{Pattern}(\sigma, w_1, v_1 v_2)$, and it is the first such division, i.e., there is no such division $w = \bar{w}_1 \bar{v}_1 \bar{v}_2 \bar{v}_3 \bar{w}_2$ where $\bar{w}_1 \bar{v}_1 \bar{v}_2 \bar{v}_3$ is a strict prefix of $w_1 v_1 v_2 v_3$.¹
- $\text{reduce}(\sigma, w_1 v_1 v_2 v_3 w_2)$, where $w_1 v_1 v_2 v_3 w_2$ is a full loop, is a sequence σ' built in two steps. First, for every path π in σ such that $\pi = R_0 \xrightarrow{w_1 v_1} R \xrightarrow{v_2} R \xrightarrow{v_3 w_2} R'$, put a path $\pi' = R_0 \xrightarrow{w_1 v_1} R \xrightarrow{v_3 w_2} R'$ into σ'' (take only paths which loop over v_2 and cut this loop out). Secondly, build the greatest sequence $\sigma' \subseteq \sigma''$ such that for all states of the region automaton $(l, D) \in \mathcal{R}(\sigma', w_1 v_1)$ there is a $\nu \in D$ such that $(\nu, \nu) \in \text{Pattern}(\sigma'', w_1 v_1, v_2)(l, D)(l, D)$.

¹ If the prefix is nonstrict then we fix any order on such divisions and pick the first one.

- $\text{extend}(\sigma, a)$ where $a \in \Sigma$ is the sequence σ' over wa such that if $\pi \in \sigma$ and π' is an extension of π by t^*a in the region automaton then $\pi \in \sigma'$ (σ' is the greatest extension of σ over wa).

The full loop concept can be seen as a partial function which returns a division such that a set of states repeats four times, creating three segments. These segments have the same reachability pattern and also their composition has this pattern. Note, that also $\text{Pattern}(\sigma, w_1, v_1v_2v_3) = \text{Pattern}(\sigma, w_1v_1, v_2v_3) = \text{Pattern}(\sigma, w_1, v_1)$. We would like to have at most one full loop for each σ , therefore we want to specify which division is chosen if there are several divisions of the required properties. We want to choose the shortest such division, i.e., we want w_2 to be as big as possible. If there are several such divisions then we fix some rule to pick one. This decision does not play any role in the correctness of the algorithm, i.e., the algorithm works for any fixed rule.

The reduction step works with the full loop of a sequence and returns another sequence. It first chooses only paths which create a loop over the middle segment from the full loop (v_2). Then it checks whether these loops (consisting of a single path) can be iterated. If yes, then the loop is cut out and the reduced path is added into the new sequence. Clearly, the sequence is state complete, because we add all paths from the original (state complete) sequence which go through these states.

The extension step just performs one more symbolic transition labeled by each letter from the alphabet (this transition can be preceded by some symbolic time delay transitions). For each path, it checks whether it can be extended by some time delay transitions and a discrete transition in the region automaton. If it is the case, then all such extensions (for any number of time delays and nondeterministic choice of the transitions) are added into the new sequence. Clearly, the new sequence is state complete.

The set of sequences together with the transitions given by the reduction and the extension function forms a transition system. The algorithm performs a standard reachability procedure in this transition system, looking for a sequence, which does not lead into any accepting state of the region automaton.

The Algorithm. Add the set containing the empty path (over an empty word) starting at R_0 into Visited.

1. Pick one sequence σ over w from Visited.
2. **Counterexample?** if $\mathcal{R}(\sigma, w)$ does not contain any accepting state, stop and answer 'NO' (the sequence of steps leading to σ gives a counterexample).
3. **Extension.** If there is no full loop of σ then remove σ from Visited, add σ to Passed, and for all $a \in \Sigma$, if $\text{extend}(\sigma, a) \notin \text{Passed}$ then add $\text{extend}(\sigma, a)$ into Visited.
4. **Reduction.** If $w = w_1v_1v_2v_3w_2$ is the full loop of σ then let $\sigma' = \text{reduce}(\sigma, w_1v_1v_2v_3w_2)$. Remove σ from Visited and if $\sigma' \notin \text{Passed}$ then add σ' into Visited.
5. **Universality?** If Visited is empty then stop and answer 'YES'. Otherwise, go to Step 1.

A situation where a sequence has a full loop is depicted in Figure 3. Before we show the correctness of the algorithm, we prove a lemma stating that if $L_\varepsilon(\mathcal{A})$ is untimed nonuniversal for all ε , but $L(\mathcal{A})$ is untimed universal, then the language $\Sigma^* - \bigcap_\varepsilon \mathcal{U}(L_\varepsilon(\mathcal{A}))$ (set of words which do not belong to some $L_\varepsilon(\mathcal{A})$) has a special structure. There is an infinite subset of this language obtained by pumping a word of a certain form.

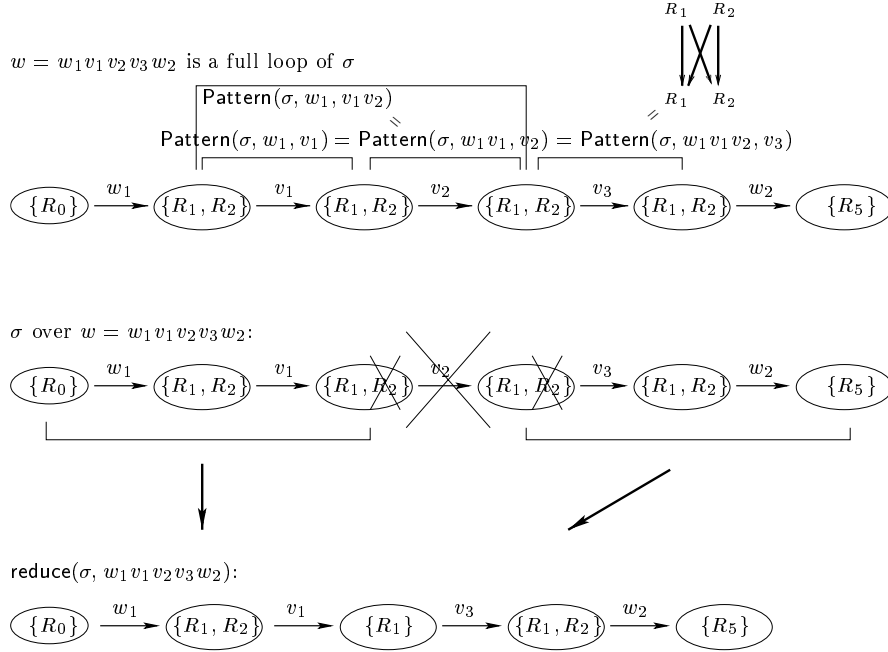


Fig. 3. The reduction step of the algorithm. There is a full loop $w = w_1 v_1 v_2 v_3 w_2$ of σ . Let us assume that there is no ν such that $(\nu, \nu) \in \text{Pattern}(\sigma, w_1, v_1)(R_2, R_2)$ (no loops over v_2 starting at R_2 can be iterated). Then $\text{reduce}(\sigma, w_1 v_1 v_2 v_3 w_2)$ will cut out all paths that go through R_2 after reading $w_1 v_1$.

The correctness of the algorithm is based on the following two facts. At first, the set of the reachable sequences is finite, because for each timed automaton \mathcal{A} there is a constant $\mathcal{H}_\mathcal{A}$ such that each sequence over w , where $|w| > \mathcal{H}_\mathcal{A}$, has a full loop. This means that it can be (and will be) shortened by the reduction step. The algorithm extends only irreducible sequences. The precise value of $\mathcal{H}_\mathcal{A}$ depending on the timed automaton parameters is shown in [2].

Secondly, there is an ε such that the timed automaton is not ε -untimed universal if and only if a sequence σ over w which does not lead into any accepting state ($\mathcal{R}(\sigma, w)$ does not contain any accepting state of the region automaton) is reachable. The rest of this section is devoted to this argument.

We state that if $L_\varepsilon(\mathcal{A})$ is untimed nonuniversal for all ε , but $L(\mathcal{A})$ is untimed universal, then the language $\Sigma^* - \bigcap_\varepsilon \mathcal{U}(L_\varepsilon(\mathcal{A}))$ (set of words which do not belong to some $L_\varepsilon(\mathcal{A})$) has a special structure. There is an infinite subset of this language obtained by pumping a word of a certain form. Also, if $L_\varepsilon(\mathcal{A})$ is untimed universal for all ε then for each word there is an accepting path of a special structure defined by the following inductive definition.

Definition 5. *Given a sequence σ over w , a path $\pi \in \sigma$ is fully accepting if it is accepting and either*

- σ, w does not have a full loop, or
- $w = w_1 v_1 v_2 v_3 w_2$ is a full loop, $\pi = R_0 \xrightarrow{w_1 v_1} R \xrightarrow{v_2} R \xrightarrow{v_3 w_2} R_f$ (denote $\bar{\pi} = R \xrightarrow{v_2} R$ the segment of π over v_2), there is a valuation ν such that $(\nu, \nu) \in C^{\bar{\pi}}$, and $\pi' = R_0 \xrightarrow{w_1 v_1} R \xrightarrow{v_3 w_2} R'$ is fully accepting over $\text{reduce}(\sigma, w_1 v_1 v_2 v_3 w_2)$ and $w_1 v_1 v_3 w_2$.

Let us for a given timed automaton \mathcal{A} and a word w denote by $\sigma_{\mathcal{A}}^w$ a set of all paths labeled by w' such that $w = w' \upharpoonright t$ in the region automaton of \mathcal{A} . This set is state complete, hence a sequence. The proofs of the following lemmata can be found in [2].

Lemma 3. *For a given timed automaton \mathcal{A} , if for all words w there is a fully accepting path over $\sigma_{\mathcal{A}}^w$ then there is an ε such that $L_\varepsilon(\mathcal{A})$ is untimed universal.*

To show the converse, namely that if there is a word w without a fully accepting path over $\sigma_{\mathcal{A}}^w$ then there is no ε such that $L_\varepsilon(\mathcal{A})$ is untimed universal, we define a recursive function **pumping** which for a given sequence, a word, and a number returns a word. If σ over w does not have a full loop then $\text{pumping}(\sigma, w, n) = w$. If $w = w_1 v_1 v_2 v_3 w_2$ is a full loop then the function creates two auxiliary words $w' = w_1 v_1 \star_{v_2} v_3 w_2$ which contains a special mark \star_{v_2} instead of the subword v_2 and $w'' = v_2^n$. In case that v_2 contained some marks, they occur in every copy of v_2 . Then it assigns to \bar{w} the result of $\text{pumping}(\text{reduce}(\sigma, w_1 v_1 v_2 v_3 w_2), w', n)$ and replaces every occurrence of \star_{v_2} in \bar{w} by w'' .

Lemma 4. *For a given timed automaton \mathcal{A} , if there is a word w such that no path $\pi \in \sigma_{\mathcal{A}}^w$ is fully accepting then for every ε there is $n \in \mathbb{N}$ such that $\text{pumped}(\sigma_{\mathcal{A}}^w, w, n) \notin L_\varepsilon(\mathcal{A})$.*

It remains to show that the counterexample produced by the algorithm corresponds to such a word with no fully accepting path and if there is such a word then the algorithm finds it.

Lemma 5 (Correctness). *For a given timed automaton \mathcal{A} , the algorithm always stops and it answers 'YES' if and only if there exists an ε such that $L_\varepsilon(\mathcal{A})$ is untimed universal.*

Proof (Sketch). If for a given timed automaton the algorithm stops and answers 'NO' then it reports a sequence of steps as a counterexample. Let us consider

the word w obtained by concatenating the letters from the `extend` operations in this sequence. Since the algorithm follows Definition 5 and reaches a sequence leading only into nonaccepting states, there is no fully accepting path in $\sigma_{\mathcal{A}}^w$. From Lemma 4, there is no ε such that $L_\varepsilon(\mathcal{A})$ is untimed universal.

If the timed automaton \mathcal{A} is not untimed universal then the algorithm stops and answers 'NO'. If \mathcal{A} is untimed universal but there is no ε such that $L_\varepsilon(\mathcal{A})$ is untimed universal then from Lemma 3 there is a word w such that no path in $\sigma_{\mathcal{A}}^w$ is fully accepting. Because the algorithm follows Definition 5, it will stop and answer 'NO'.

5 Existential Sampled Universality for Infinite Words

In this section, we give the positive answer to the decidability of the problem whether there exists an ε such that the untimed ε -sampled ω -language is universal.

Theorem 2. *For a given timed automaton \mathcal{A} , the question whether there exists an ε such that $L_\varepsilon^\omega(\mathcal{A})$ is untimed universal is decidable.*

We show that the algorithm for finite words can be easily modified to work also for infinite words. This algorithm resembles the construction for the complementation of Büchi automata used by Sistla, Vardi, and Wolper [19]. For a given timed automaton \mathcal{A} , $L_\varepsilon^\omega(\mathcal{A})$ is clearly untimed nonuniversal, if $L_\varepsilon(\mathcal{A}')$ is not untimed universal, where \mathcal{A}' is obtained from \mathcal{A} by changing all locations to accepting. Otherwise, we need to find a word w_1w_2 such that there is no ε such that there is a run of \mathcal{A} over $w_1w_2^\omega$ in ε -sampled semantics. But for this is it sufficient to find a sequence σ over some w_1w_2 reachable by the algorithm such that $\mathcal{R}(\sigma, w_1) = \mathcal{R}(\sigma, w_1w_2) = \mathcal{R}(\sigma, w_1w_2^2)$, $\text{Pattern}(\sigma, w_1, w_2) = \text{Pattern}(\sigma, w_1, w_2^2)$, and no loop $R \xrightarrow{w_2} R$ from σ contains an accepting state. We would like to find a condition which would make the following claim hold. There is a path $R \xrightarrow{w_2} R$ for some $R \in \mathcal{R}(\sigma, w_1)$ containing an accepting state if and only if $w_1w_2^\omega \in L(\mathcal{A})$. Such a condition is that there is a path $R \xrightarrow{w_2} R'$ if and only if there is R'' such that there are paths $R \xrightarrow{w_2} R''$ and $R \xrightarrow{w_2} R'$. But this is implied by $\text{Pattern}(\sigma, w_1, w_2) = \text{Pattern}(\sigma, w_1, w_2^2)$.

Therefore, it is enough to modify the condition for answering 'NO' and reporting a counterexample in the following way. For the examined sequence σ over w , there is a division of w into w_1w_2 such that $\mathcal{R}(\sigma, w_1) = \mathcal{R}(\sigma, w_1w_2) = \mathcal{R}(\sigma, w_1w_2^2)$, $\text{Pattern}(\sigma, w_1, w_2) = \text{Pattern}(\sigma, w_1, w_2^2)$, and no loop $R \xrightarrow{w_2} R$ from σ contains an accepting state.

Lemma 6 (Correctness). *For a given timed automaton \mathcal{A} , the modified algorithm always stops and it answers 'YES' if and only if there exists an ε such that $L_\varepsilon^\omega(\mathcal{A})$ is untimed universal.*

6 Universal Sampled Universality

A dual question to the one studied in the previous subsection is whether for all ε it holds that $L_\varepsilon(\mathcal{A})$ respectively $L_\varepsilon^\omega(\mathcal{A})$ is timed universal. We show that these questions are undecidable. The finite word case is equivalent to the dense time universality and a technique from [1] applies for the infinite word case.

Theorem 3. *For a given timed automaton \mathcal{A} , the question whether for all ε it holds that $L_\varepsilon(\mathcal{A})$ ($L_\varepsilon^\omega(\mathcal{A})$) is timed universal is undecidable.*

Proof. In the case of finite timed traces, we show that the timed universality of $L_\varepsilon(\mathcal{A})$ is equivalent to the timed universality of $L(\mathcal{A})$, which has been proved undecidable in [3]. Assume that the answer for the sampled universality problem for a given automaton \mathcal{A} is 'NO'. Then $L(\mathcal{A})$ is not dense time universal, because for each timed trace $w \notin L_\varepsilon(\mathcal{A})$ it holds that $w \notin L(\mathcal{A})$. To show the other direction, assume that for a given timed automaton \mathcal{A} the language $L(\mathcal{A})$ is not timed universal. Then there is a timed trace w such that $w \notin L(\mathcal{A})$ and all timestamps are rational. This means that all runs of the automaton \mathcal{A} over this timed trace are nonaccepting. But then there is an ε for which this timed trace is also a \mathcal{T}_ε -timed trace (and also not accepted).

In the case of infinite timed traces, this argument does not work, because an infinite timed trace with rational timestamps violating universality in the dense time case does not have to be a \mathcal{T}_ε -timed trace for any ε .

Due to Mayr [14], the existence of a space-bounded recurrent-state computation with insertion errors for alternating channel machines is undecidable. We sketch an adaptation of the reduction of this problem to the universality checking for one clock Büchi timed automata from [1]. We need to build a timed automaton \mathcal{A} for every alternating channel machine M such that M has a space-bounded recurrent-state computation with insertion errors if and only if for all ε , $L_\varepsilon^\omega(\mathcal{A})$ is timed universal.

The proof in [1] specifies five conditions in Definition 4 for an ω -language so that its words correspond exactly to space-bounded recurrent-state computations with insertion errors of M . We build \mathcal{A} such that it accepts precisely those words that fail to satisfy any of the conditions 1–4 (there is such an automaton even with one clock). The last condition from this definition says that the maximal number of the events in any time unit is bounded. But if there is an ε such that $L_\varepsilon^\omega(\mathcal{A})$ is not timed universal then it means that the words accepted by \mathcal{A} in ε -sampled semantics also satisfy this condition.

Conversely, if M has no space-bounded recurrent-state computation with insertion errors then each \mathcal{T}_ε -timed trace must violate one of the conditions 1–4, because it automatically satisfies the condition 5. Therefore, it is accepted by \mathcal{A} .

7 Remaining Variants

There are two other decision problems arising naturally in our scheme. Both of them are decidable, but as we show, the problems degenerate to checking (un)timed universality of $L_\varepsilon(\mathcal{A})$ or $L_\varepsilon^\omega(\mathcal{A})$ for one fixed ε , namely $\varepsilon = 1$.

The first problem is to decide whether there is an ε for a timed automaton \mathcal{A} such that $L_\varepsilon(\mathcal{A})$ is timed universal. But if this is not true for $\varepsilon = 1$ then it is not true for any $\varepsilon < 1$. If there is a \mathcal{T}_1 -timed trace for which there is no run of \mathcal{A} then this trace is also a \mathcal{T}_ε -timed trace for every $\varepsilon < 1$ and there is no run of \mathcal{A} over this trace. The same reasoning applies also for $L_\varepsilon^\omega(\mathcal{A})$.

The other problem is to decide whether for a timed automaton \mathcal{A} it is true that for all ε , $L_\varepsilon(\mathcal{A})$ (or $L_\varepsilon^\omega(\mathcal{A})$) is untimed universal. But if this is true for $\varepsilon = 1$ then it is true for any $\varepsilon < 1$. If there is a word w for which there is an accepting run of \mathcal{A} in 1-sampled semantics then this run also accepts w in ε -sampled semantics for every $\varepsilon < 1$.

Therefore, for both cases, it is enough to check whether $L_1(\mathcal{A})$ is (un)timed universal or whether $L_1^\omega(\mathcal{A})$ is (un)timed universal and this gives us an answer for all ε or constitutes a witness of existence of an ε with the checked property.

8 Conclusions and Future Work

In this paper, we have studied the universality problems of timed automata in sampled semantics. We have shown that the question whether for all sampling periods ε a timed automaton accepts all timed words in ε -sampled semantics is undecidable. As a main result, we have presented a novel proof for the decidability of checking whether there is a sampling period such that a timed automaton accepts all untimed words in ε -sampled semantics is decidable. We believe that the proof techniques may be used to study other properties of timed systems, in particular the implementability of timed automata. As future work, we plan to extend our results to language inclusion checking, i.e, the problem: given timed automata A and B , whether there exists a sampling period ε such that $L_\varepsilon(A) \subseteq L_\varepsilon(B)$ and the related question for the untimed case. We shall also study the corresponding questions within the context of timed and untimed bisimulation.

Acknowledgments. We thank Radek Pelánek for his comments on previous drafts of this paper.

References

- [1] Parosh Aziz Abdulla, Johann Deneux, Joël Ouaknine, and James Worrell. Decidability and complexity results for timed automata via channel machines. In *Proc. of ICALP'05*, volume 3580 of *LNCS*, pages 1089–1101. Springer, 2005.
- [2] Parosh Aziz Abdulla, Pavel Krčal, and Wang Yi. Sampled universality of timed automata. Technical Report 2006-052, IT Department, Uppsala University, Dec 2006.
- [3] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.
- [4] E. Asarin, M. Bozga, A. Kerbrat, O. Maler, A. Pnueli, and A. Rasse. Data-structures for the verification of timed automata. In *Proc. of HART'97*, pages 346–360. Springer, 1997.

- [5] E. Asarin, O. Maler, and A. Pnueli. On discretization of delays in timed automata and digital circuits. In *Proc. of CONCUR'98*, volume 1466 of *LNCS*, pages 470–484. Springer, 1998.
- [6] D. Beyer. Improvements in BDD-based reachability analysis of timed automata. In *Proc. of FME 2001*, volume 2021 of *LNCS*, pages 318–343. Springer, 2001.
- [7] M. Bozga, O. Maler, and S. Tripakis. Efficient verification of timed automata using dense and discrete time semantics. In *Proc. of Charme'99*, volume 1703 of *LNCS*, pages 125–141. Springer, 1999.
- [8] H. Comon and Y. Jurski. Timed automata and the theory of real numbers. In *Proc. of CONCUR'99*, volume 1664 of *LNCS*, pages 242–257. Springer, 1999.
- [9] C. Dima. Computing reachability relations in timed automata. In *Proc. of LICS'02*. IEEE Computer Society Press, 2002.
- [10] T. A. Henzinger, Z. Manna, and A. Pnueli. What good are digital clocks? In *Proc. of ICALP'92*, volume 623 of *LNCS*, pages 545–558. Springer, 1992.
- [11] Thomas A. Henzinger and Jean-Francois Raskin. Robust undecidability of timed and hybrid systems. In *HSCC'00*, pages 145–159. London, UK, 2000. Springer.
- [12] P. Krčál and R. Pelánek. On sampled semantics of timed systems. In *Proc. of FSTTCS'05*, volume 3821 of *LNCS*, pages 310–321. Springer, 2005.
- [13] K. G. Larsen and W. Yi. Time-abstracted bisimulation: Implicit specifications and decidability. *Information and Computation*, 134(2):75–101, 1997.
- [14] R. Mayr. Undecidable problems in unreliable computations. *Theoretical Computer Science*, 297:347–354, 2003.
- [15] J. Ouaknine and J. Worrell. Revisiting digitization, robustness, and decidability for time d automata. In *Proc. of LICS'03*, pages 198–207. IEEE Computer Society Press, 2003.
- [16] Joël Ouaknine and James Worrell. Universality and language inclusion for open and closed timed automata. In *Proc. of HSCC'03*, volume 2623 of *Lecture Notes in Computer Science*, pages 375–388. Springer, 2003.
- [17] Joël Ouaknine and James Worrell. On the language inclusion problem for timed automata: Closing a decidability gap. In *Proc. of LICS 2004*, pages 54–63. IEEE Computer Society, 2004.
- [18] Anuj Puri. Dynamical properties of timed automata. *Discrete Event Dynamic Systems*, 10(1-2):87–113, 2000.
- [19] A. Prasad Sistla, M. Y. Vardi, and P. Wolper. The complementation problem for Büchi automata with applications to temporal logic. *Theoretical Computer Science*, 49(2-3):217–237, 1987.
- [20] M. De Wulf, L. Doyen, and J.-F. Raskin. Almost ASAP semantics: From timed models to timed implementations. In *Proc. of HSCC'04*, volume 2993 of *LNCS*, pages 296–310. Springer, 2004.