Almost Periodicity and the Timestamp of Timed Automata

Amnon Rosenmann

AIT Austrian Institute of Technology GmbH
Vienna, Austria
amnon.rosenmann@gmail.com

Abstract. Given a non-deterministic timed automaton with silent transitions (eNTA), we show that after an initial stage it becomes time-periodic. After computing the periodic parameters, we construct a finite almost periodic augmented region automaton, which includes a clock measuring the global time. In the next step we construct the timestamp of the automaton: the union of all its observable timed traces, which contains all the dates on which events occur - a generalization of the reachability problem. The timestamp of each event is an almost periodic subset of the non-negative reals. We also construct a simple deterministic timed automaton with the same timestamp as the given timed automaton, in contrast to the fact that the timed automaton itself may be non-determinizable. One application is the decidability of the 1-bounded language inclusion problem for eNTA. Another is a partial method, which is not bounded by time or number of steps, for showing the non-inclusion of languages of timed automata.

1 Introduction

A timed automaton (TA) is an abstract model of real-time systems in the form of an extension of a finite automaton with time properties. The class DTA of deterministic timed automata is strictly included in the class NTA of non-deterministic timed automata [10], and the latter is strictly included in the class eNTA of non-deterministic timed automata with silent transitions [7]. Consequently, one of the basic questions, whether the language accepted by one TA (e.g. the implementation) is included in the language accepted by another TA (e.g. the specification) is undecidable for NTA but decidable for DTA. On the other hand, for special sub-classes or modifications it was shown that decidability exists (see [2,8,12,11,8,4] for a partial list).

In this note we look at the language of a non-deterministic timed automaton with silent transitions from a ‘weaker’ perspective. Instead of referring to the set of all observable timed traces we examine the union of all these timed traces. Our motivating goal was to provide an answer to the question:

On which dates can an event of the timed automaton occur?
This is a generalization of the reachability problem \cite{1, 3}. We remark that there are variants of the reachability problem, e.g. in timed game automaton (see \cite{K}).

We consider here the set (of the continuum cardinality) of all runs of an eNTA, and it does not matter whether we look only at finite runs or at infinite (countable) runs. The union of the records on the real time line of the events of all runs, which we call the timestamp of the timed automaton, has an almost periodic pattern, that is, except for a possible prefix, it is periodic. Both the prefix part and the part that repeats itself are composed of finitely-many integral points or intervals of integral end-points (or an infinite ray). The timestamp can be represented by a simple deterministic timed automaton that can be algorithmically computed. In fact, non-determinism and silent transitions do not impose any theoretical or algorithmical difficulties or complexity on what is presented in this note.

Questions like universality or language inclusion that are undecidable in general at the level of non-deterministic timed automata become decidable when examining the analog notions with respect to the timestamp of timed automata.

Clearly, when the language of one timed automaton is included in the language of another automaton then so is the case also with their timestamps, but the other way round does not necessarily hold.

The paper is organized in the following way. In Section 2 basic definitions concerning timed automata are given. In Section 3 we present a lazy description of the trail and timestamp of a single path of a timed automaton. The description is more from a geometric than algebraic point of view and it resembles the known decomposition of the trail into zones and regions. The novelty is in the introduction of the global time \(t\) as part of the clocks of the system, although the analysis itself is standard since the behaviour of \(t\) does not deviate from the behaviour of the regular clocks. Here, and in what follows, we ignore complexity issues since our purpose is merely to demonstrate the existence of certain patterns in TA without trying to find efficient algorithms.

Our aim is find some time-periodic structure in an eNTA \(A\) and to construct a finite version of it which admits this periodic structure. In order to make the analysis simpler, we work with the region automaton. The problem is that when trying to add \(t\) to the region automaton, we are confronted with the problem that since \(t\) never resets we end up with infinitely-many regions. To overcome this obstacle, we ignore the integral part of \(t\) and leave only its fractional part. The result is the augmented region automaton \(\mathcal{R}(A)\) (Section 4). The compensation for the partial information concerning \(t\) comes in the form of integral time labels on the edges. If needed, one could also add the clocks constraints to the edges in order to maintain the same information as in \(A\).

The augmented region automaton still does not reveal the time-periodic structure. Hence, we first unfold it and construct the infinite augmented region automaton \(\mathcal{R}_\infty(A)\) (Section 5). Then, in Section 6 we explore both \(\mathcal{R}(A)\) and \(\mathcal{R}_\infty(A)\) and find the periodic structure. Two time parameters are computed: one parameter, \(t_{\text{per}}\), tells when the a-periodic part is already passed, and the
other, \( L \), gives a time-period. We do not try to find the minimal values of these parameters although this is possible.

The meaning here of ‘time-periodic’ is that after some finite time the structure of \( A \) becomes periodic. It is not that every timed trace becomes periodic, but the following holds. Let \( r \) be a run of \( A \) and let \( s \) be the suffix of \( r \) that occurs after the value of \( t \) passed \( t_{\text{per}} \). Then there are infinitely-many runs of \( A \) with the same suffix \( s \), only shifted in time by multiples of \( L \) (assuming that \( A \) is not bounded in time). Given the periodic structure it is now easy to fold \( R_{t_{\text{per}}}^T(A) \) into a finite automaton, the almost periodic augmented region automaton \( R_{t_{\text{per}}}^T(A) \) (Section 7), which reveals the time-periodic structure.

We started by exploring the timestamp of a single path of \( A \). In the last section (Section 8) we close the loop by constructing the entire timestamp of \( A \), which is an almost periodic subset of \( \mathbb{R}_{\geq 0} \). We also show that the 1-bounded universality and language inclusion problems are decidable for eNTA. As for the general language inclusion problem in eNTA, the timestamp, or better the more informative \( R_{t_{\text{per}}}^T \) construct, may serve mainly for demonstrating the non-inclusion relation between the languages of two eNTAs.

### 2 Timed Automata with Silent Transitions

We start with some definitions and notation. A timed automaton (TA) is an abstract model aiming at capturing the temporal behaviour of systems. It is a finite automaton extended with a set of clocks defined over \( \mathbb{R}_{\geq 0} \), the set non-negative real numbers.

We may represent the timed automaton by a graph whose nodes are called locations. While being at a location, all clocks progress at the same rate. The edges of the graph are called transitions. Each transition may be subject to some constraints, called guards, put on clocks values in the form of integer inequalities. At each such transition an event (or action) occurs and some of the clocks may be reset. There exists a distinction between observable events, that can be traced by an outside observer, and silent events, that are inner and non-observable ones. The finite set of different labels of observable events is denoted by \( \Sigma \), whereas each silent event is represented by the same label \( \epsilon \).

Let \( C \) be a finite set of clock variables. A clock valuation \( v(c) \) is a function \( v : C \to \mathbb{R}_{\geq 0} \) assigning a real value to every clock \( c \in C \). We denote by \( V \) the set of all clock valuations and by \( 0 \) the valuation assigning 0 to every clock. For a valuation \( v \) and \( d \in \mathbb{R}_{\geq 0} \) we define \( v + d \) to be the valuation \( (v + d)(c) := v(c) + d \) for all \( c \in C \). For a subset \( C_{\text{rst}} \) of \( C \), we denote by \( v|_{C_{\text{rst}}} \) the valuation such that for every \( c \in C_{\text{rst}} \), \( v|_{C_{\text{rst}}}(c) = 0 \) and for every \( c \in C \setminus C_{\text{rst}} \), \( v|_{C_{\text{rst}}}(c) = v(c) \). A clock constraint \( \varphi \) is a conjunction of predicates of the form \( c \sim n \), where \( c \in C, n \in \mathbb{N}_0 := \mathbb{N} \cup \{0\} \) and \( \sim \in \{<,\leq,=,\geq,>\} \). Given a clock valuation \( v \), we write \( v \models \varphi \) when \( v \) satisfies \( \varphi \).

We give now a formal definition of timed automata with silent transitions.

**Definition 1** (eNTA). A non-deterministic timed automaton with silent transitions \( A \) is a tuple \((Q,q_0,\Sigma,\mathcal{C},T)\), where
1. $Q$ is a finite set of locations and $q_0$ is the initial location;
2. $\Sigma_e = \Sigma \cup \{\epsilon\}$ is a finite set of event labels, where $\Sigma$ are the observable events and $\epsilon$ represents a silent event;
3. $C$ is a finite set of clock variables;
4. $T \subseteq Q \times \Sigma_e \times G \times P(C)$ is a finite set of transitions of the form $(q, a, g, C_{rst}, q')$, where
   (a) $q, q' \in Q$ are the source and the target locations respectively;
   (b) $a \in \Sigma_e$ is the event label;
   (c) $g \in G$ is the transition guard, which is a conjunction of constraints of the form $c \sim n$, where $c \in C$, $\sim \in \{<, \leq, =, \geq, >\}$ and $n \in \mathbb{N}_0$;
   (d) $C_{rst} \subseteq C$ is the subset of clocks to be reset;

Remark: it is common to include in the definition of a timed automaton a set of location invariants, where each invariant is a conjunction of constraints of the form $c < n$, $n \in \mathbb{N}$, or $c \leq n$, $n \in \mathbb{N}_0$, $c \in C$. But for the purpose of this note these constraints are not needed and we may consider them to be more of a 'syntactic sugar' because they may be incorporated in the transition guards such that the guards never pass the source location invariants and the target location invariants (after resetting the appropriate clocks in the transition). Notice also that for simplicity we mostly do not distinguish here between 'accepting' and non-accepting locations (we may think of all locations as accepting), as this distinction does not alter the results.

Semantics of eNTA. The semantics of an eNTA $A$ is given by the timed transition system $[[A]] = (S, s_0, \mathbb{R}_{\geq 0}, \Sigma_e, T)$, where

1. $S = \{(q, v) \in Q \times V\}$
2. $s_0 = (q_0, 0)$
3. $T \subseteq S \times (\Sigma_e \cup \mathbb{R}_{\geq 0}) \times S$ is the transition relation consisting of timed and discrete transitions, such that:
   (a) Timed transitions (delays): $((q, v), d, (q, v + d)) \in T$, where $d \in \mathbb{R}_{\geq 0}$;
   (b) Discrete transitions (jumps): $((q, v), a, (q', v')) \in T$, where $a \in \Sigma$ and there exists a transition $(q, a, g, C_{rst}, q')$ in $T$, such that $v \models g$ and $v' = v|_{C_{rst}}$.

A (finite) run $r$ of an eNTA $A$ is a sequence of alternating timed and discrete transitions of the form

$$(q_0, 0) \xrightarrow{d_1} (q_0, 0 + d_1) \xrightarrow{\tau_1} (q_1, v_1) \xrightarrow{d_2} \cdots \xrightarrow{d_k} (q_{k-1}, v_{k-1} + d_k) \xrightarrow{\tau_k} (q_k, v_k),$$

where $\tau_i = (q_{i-1}, a_i, g_i, C_{rst(i)}, q_i) \in T$ and $a_i \in \Sigma$. The run $r$ of $A$ induces the timed trace

$$\sigma = (t_1, a_1), (t_2, a_2), \ldots, (t_k, a_k),$$

with $a_i \in \Sigma_e$ and $t_i = \sum_{j=1}^{i} d_i$. From the latter we can extract the observable timed trace, which is obtained by deleting from $\sigma$ all the pairs containing silent events.
We remark that when the focus is on the language of a timed automaton then one distinguishes between finite and infinite timed traces, but in our case the results are the same whether we consider finite or infinite runs and timed traces.

A TA is called deterministic if it does not contain silent transitions, it contains one initial location and whenever two timed traces are the same they induce the same run. Otherwise, the TA is non-deterministic.

3 The Trail and Timestamp of a Single Path

In this section we explore the trail and timestamp of a single path of a timed automaton. Given a timed automaton \( A \in \text{eNTA} \), we add to it the clock \( t \), which measures the global time. The value of \( t \) at the initial state is 0 and then it advances without reset throughout each run of \( A \), with all other clocks advancing at the same rate.

3.1 Single Clock

We begin with a simple automaton \( A \) having a single clock \( x \) (in addition to \( t \)). Let \( \gamma \) be a finite path in the graph of \( A \), and suppose that the set of possible runs along \( \gamma \) is not empty. For each run \( r \) along \( \gamma \), its trajectory, which consists of all points \((t, x)\) of \( r \) in the \( t \times x \) plane, forms a piecewise continuous 'function' with each continuous segment being a sloped line. The union of all the trajectories over all possible runs \( r \) along \( \gamma \), denoted \( \rho_\gamma \), is the trail of \( \gamma \).

In order to draw \( \rho_\gamma \) and the timestamps of the events along \( \gamma \) we will process all runs simultaneously. At the initial state \( x \) is reset to zero and then (until \( x \) is reset again) the trail \( \rho_\gamma \) is in the form of a straight line of slope 1 that starts at \((0, 0)\), as seen in Fig. 1.

In fact, at times \( t \) where \( x \) is reset we generally have two values of \( x \) for the same value of \( t \).

![Fig. 1. Timestamp of two events (single clock)](image)
If an event occurs when the value of clock $x$ is $n$, $n \in \mathbb{N}_0$, we draw a horizontal line at level $n$ in the $t \times x$ plane, check where the horizontal line meets the trail, and then draw a vertical line at the point of intersection to obtain the timestamp of the event. Clearly, the latter is also of an integral value (see event and timestamp 1 at time $t = 1$ in Fig. 1). If $x$ is not reset during an event then the trail continues to be a straight line.

If, on the other hand, $x$ is reset then the trail is cut and starts again from the $t$-axis. In the case of an event which is a singleton then the trail continues in a straight line parallel to its starting line (as after event 1 in Fig. 1). Otherwise, if an event occurs when e.g. $n < x < n + 1$ then the infinite horizontal stripe that represents this constraint intersects the trail in a diagonal segment and the timestamp, the orthogonal projection to the $t$-axis, forms an open time interval with integral endpoints (see event and timestamp 2 at time $2 < t < 3$ in Fig. 1). If now, $x$ is also reset on this event then the trail is cut and continues in the form of a parallelogram that starts at the horizontal segment on the $t$-axis which refers to the timestamp (see the trail in Fig. 2 after event 2).

The timestamps of next events will not be single points anymore but intervals. Moreover, they will always be bounded between integral points (see e.g. event 3 in Fig. 2 that occurs when $1 < x < 2$ with a timestamp $3 < t < 5$ of size 2, or event 4 that happens when $x = 4$ and with timestamp the interval $(6, 7)$), unless the transition constraint is not bounded from above and in this case the timestamp of the event extends to infinity.

Another characteristic of a single clock is that the width of the trail never decreases (this is seen Fig. 2 and if clock $x$ was reset at event 3 then the width of the trail would have been increased to 2).

We can summarize our observations so far in the following propositions, which are easily proven based on the fact that each trajectory is a line of slope 1.

![Diagram showing timestamps and events for single clock](image)
Proposition 1. The trail of a path $\gamma$ in the eNTA $A$ consists of (possibly unbounded) parallelograms (stripes) of integral horizontal width (in the finite case) and of slope 1, where the base of each parallelogram lies on the $t$-axis between some points $m$ and $n$, $m \leq n$, $m \in \mathbb{N}_0$ and $n \in \mathbb{N}_0 \cup \infty$.

Proposition 2. The timestamp of each event involving a single clock is either an integral point or an (open, closed or half-open) interval between points $m$ and $n$, $m < n$, $m \in \mathbb{N}_0$ and $n \in \mathbb{N}_0 \cup \infty$.

In the next proposition we show how to compute the non-decreasing sequence of the widths of the trail. Given a path $\gamma$ in $A$, we denote by $d_i^\gamma$ the actual duration of time at which the $i$-th event along $\gamma$ can occur, and it is computed in the following way. Let $l_i^\gamma, u_i^\gamma$ be the lower and upper bound of the guard on $x$ at the $i$-th transition along $\gamma$ ($l_i^\gamma = u_i^\gamma$ in case of an exact time event). When the lower bound is missing it is set to 0, and when an upper bound is missing it is set to $\infty$. Then, if $x$ is not reset at event $i$ and $l_{i+1}^\gamma < l_i^\gamma$ then we set $l_{i+1}^\gamma := l_i^\gamma$, and if $u_i^\gamma > u_{i+1}^\gamma$ then we set $u_{i}^\gamma := u_{i+1}^\gamma$. We perform this series of assignments starting from the last transition along $\gamma$ and moving backward. We define $d_i^\gamma$ to be $d_i^\gamma := u_i^\gamma - l_i^\gamma$. Let also $s_i^\gamma$ be the size of the timestamp of the $i$-th event, and let $w_i^\gamma$ be the width of the trail after the $i$-th event, with $w_0^\gamma$ the initial width of the trail (the width is the size of the intersection of an horizontal line with the trail).

Definition 2. A path in $A$ is called reachable if the set of runs along the path is non-empty.

Proposition 3. Let $\gamma$ be a reachable path in $A$. Then

$$s_i^\gamma = w_{i-1}^\gamma + d_i^\gamma,$$

with $w_0^\gamma = 0$, and for $i > 0$,

$$w_i^\gamma = \begin{cases} w_{i-1}^\gamma & \text{if } x \text{ is not reset on event } i \\ s_i^\gamma & \text{if } x \text{ is reset on event } i. \end{cases}$$

Proof. When $x$ is not reset on an event then the trail continues as before the event and certainly its width does not change: $w_i^\gamma = w_{i-1}^\gamma$. If $x$ is reset then the size of the timestamp of the event is the size of the interval in the $t$-axis which consists of all points $(t,0)$ of the trail after the reset. But after the event the trail contains all the parallel trajectories that pass through these points $(t,0)$ and so the new width of the trail is $w_i^\gamma = s_i^\gamma$.

Let us look now at the size of the timestamp of the $i$-th event. Suppose it refers to the actual constraint $m < x < m + d_i^\gamma$ (the strict inequalities may be replaced by weaker ones). Let $t_0 = \inf\{(t,m) \mid (t,m) \text{ in the trail}\}$. Then the event refers to the parallelogram (or to a line, in the degenerate case) with vertices at $(t_0,m), (t_0 + w_{i-1}^\gamma, m), (t_0 + d_i^\gamma, m + d_i^\gamma), (t_0 + w_{i-1}^\gamma + d_i^\gamma, m + d_i^\gamma)$, and the earliest time at which the event can occur is $t_0$, while the latest time is $t_0 + w_{i-1}^\gamma + d_i^\gamma$. Thus, $s_i^\gamma = w_{i-1}^\gamma + d_i^\gamma$. \qed
In order to make the analysis in the case of multiple clocks simpler, we partition the time domain, both that of $t$ and that of $x$, into the subsets of integral points and open unit intervals of integral end-points: \{0\}, \{0, 1\}, \{1\}, \{1, 2\}, \{2\}, etc., and this partition is carried-out also to the constraints of the transition guards of the timed automaton. Then, by intersecting the trail $\rho_\gamma$, which is in the form of a diagonal stripe, with the horizontal and vertical lines, or stripes of width 1, we get the (indecomposable) basic shapes as seen in Fig. 3, where all shapes, except for the point, are open sets (do not include their boundaries).

The trail itself can now be decomposed into diagonal lines and parallelograms of width 1, and each diagonal line or parallelogram can be further decomposed into basic shapes, as seen in Fig. 4. Let us denote by $\{z\}$ the fractional part of a real-valued number $z$. If we look at the 1-dimensional trail in Fig. 3 then we see that it represents the case where $\{x\} = \{t\}$, alternating between shape (a), in which $\{x\} = 0$, and shape (b), in which $0 < \{x\} < 1$. The 2-dimensional trail has period of length 4 consisting of the shapes (c), (d), (e) and (f) (in that order). Shape (c) represents the state where $0 = \{x\} < \{t\} < 1$. Shape (d) represents the state where $0 < \{x\} < \{t\} < 1$. Shape (e) represents the state where $0 = \{t\} < \{x\} < 1$. Finally, shape (f) represents the state where $0 < \{t\} < \{x\} < 1$.

3.2 Multiple Clocks

Suppose now that the timed automaton $A$ has $r$ clocks $x_1, \ldots, x_r$. Given a fixed path $\gamma$ on the graph of $A$, the trail $\rho_\gamma$ of $\gamma$ consists now of all points $(t = x_0, x_1, \ldots, x_r)$ that can be reached in any run through $\gamma$. The trail forms a domain of dimension at most $r + 1$ in the $(r + 1)$-dimensional Euclidean space. For each clock $x_i, i > 0$, the projection of $\rho_\gamma$ to the $t \times x_i$ plane is then similar to the picture we got for an automaton with a single clock.

Again, in order to make the analysis simpler, we partition the trail into simplicial trails, which are (possibly unbounded) parallelotopes of width 1 or 0 in each dimension, and then we further triangulate each simplicial trail into basic domains. The dimension of a simplicial trail is recorded in every point $(x_0, x_1, \ldots, x_r)$ of it and is the number of different values among the fractional parts $\{x_0\}, \{x_1\}, \ldots, \{x_r\}$, a consequence of the fact that all clocks advance at the same rate (for the case where $r = 1$ we can see the 1 and 2-dimensional simplicial trails in Fig. 4).
The basic domains that form the triangulation of a simplicial trail are simplices of dimension 0 (a point), 1 (a line), 2 (a triangle), 3 (a tetrahedron) or higher, up to dimension \( r + 1 \). Each such simplex of positive dimension is an open domain that resides inside an \((r+1)\)-dimensional unit hyper-cube and with the vertices of the simplex in the lattice \( \mathbb{N}^{r+1}_0 \). A simplex is characterized by the fractional values of the clock variables, such that for all points in the simplex a fixed ordering of the form \( 0 \preceq_1 \{ x_{i_1} \} \preceq_2 \{ x_{i_2} \} \preceq_3 \cdots \preceq_r \{ x_{i_r} \}, \preceq_i \in \{=,<\} \), holds.

The decomposition of the simplicial trails into simplices is done in the following way. We start with a simplex \( n + \Delta \), which is obtained after an event, where \( n \in \mathbb{N}^{r+1}_0 \) is the vector of integral parts of the clocks, and \( \Delta \) - the fractional parts. The simplicial trail \( S \) that emerges from \( n + \Delta \) is obtained by moving \( \Delta \) along the directional vector \( \mathbf{1} = (1, 1, \ldots, 1) \), that is,

\[
S = n + \Delta + \{ \lambda \mathbf{1} | \lambda \geq 0 \},
\]

until some clock is reset during an event. The directional vector \( \mathbf{1} \) represents the fact that all clocks (including \( t \)) are progressing at the same rate, and the simplicial trail is what is referred to as the ‘closure’ of \( n + \Delta \) in Fig. 4.

The discretization of a \( k \)-dimensional simplicial trail \( S \) is done by the triangulation of \( S \) into simplices, where we switch alternately between an open \( k \)-simplex (a simplex of dimension \( k \), \( 1 \leq k \leq r + 1 \)), then a \((k-1)\)-face of the \( k \)-simplex (which is a simplex of dimension \( k - 1 \)). This \((k-1)\)-simplex is then the face of another \( k \)-simplex, then we switch to a \((k-1)\)-face of the latter, and so on (in Fig. 4 we can see it for the \( 1 \)-simplex of type (b) and its \( 0 \)-face, a point of type (a), or the \( 2 \)-simplex of type (d), then the \( 1 \)-face of type (e), then the \( 1 \)-face of type (c), and so on).

The switching from a \( k \)-dimensional simplex \( n_1 + \Delta_1 \) to its \((k-1)\)-face \( n_2 + \Delta_2 \) is done in the following way. Suppose that the clock \( x_i \) has the largest fractional
part in $\Delta_1$. Then we project $n_1 + \Delta_1$ in direction 1 to its face $n_2 + \Delta_2$ in which $\{x_i\} = 0$: each point $p_1 = (x_0, x_1, \ldots, x_i, \ldots, x_r) \in n_1 + \Delta_1$ is projected to the point $p_2 = p_1 + (1 - \{x_i\}) 1 \in n_2 + \Delta_2$ (in $n_2$ the integral value of $x_i$ is increased by 1 while the other values are the same as in $n_1$). Thus, the fractional parts of the clocks are cyclically shifted, so that $\{x_i\}$ is now at the bottom of the ordered list. Next, we switch to the $k$-simplex $n_3 + \Delta_3$, with $\{x_i\}$ becoming positive while the ordering on the fractional parts of the clocks is unchanged. Then, we project to the face of $n_3 + \Delta_3$ with $\{x_j\} = 0$, where $\{x_j\}$ is at the top of the ordered list of fractional parts (if $\{x_j\} = k_0$ then the projection is to face with $\{x_j\} = \{x_k\} = 0$), and so on.

Let us demonstrate what happens during a clock reset by an example.

**Example 1.** Suppose that there are 4 clock variables: $x_0, x_1, x_2, x_3$, and $x_2$ is reset during an event. We will examine the following cases with respect to the simplex at which the event occurs.

1. The simplex is $n_1 + \Delta_1$,
   \[ \Delta_1 = [0 < \{x_0\} < \{x_1\} < \{x_2\} < \{x_3\} < 1] \]
   of dimension 4. Each point $(x_0, x_1, x_2, x_3) \in n_1 + \Delta_1$ is projected (orthogonal projection to the hyperplane $x_3 = 0$) to the point $(x_0, x_1, 0, x_3)$ and the trail is cut and then continues from the simplex of the form $n_2 + \Delta_2$,
   \[ \Delta_2 = [0 = \{x_2\} < \{x_0\} < \{x_1\} < \{x_3\} < 1] \]
   of dimension 3. This simplex is the face of the next 4-simplex in the trail: $n_3 + \Delta_3$,
   \[ \Delta_3 = [0 < \{x_2\} < \{x_0\} < \{x_1\} < \{x_3\} < 1]. \]

2. The simplex is $n_1 + \Delta_1$,
   \[ \Delta_1 = [0 = \{x_0\} < \{x_1\} < \{x_2\} < \{x_3\} < 1] \]
   of dimension 3. Then, after the cut the simplex is of the form
   \[ \Delta_2 = [0 = \{x_0\} = \{x_2\} < \{x_1\} < \{x_3\} < 1] \]
   of dimension 2, and then
   \[ \Delta_3 = [0 < \{x_0\} = \{x_2\} < \{x_1\} < \{x_3\} < 1] \]
   of dimension 3.

3. The simplex is $n_1 + \Delta_1$,
   \[ \Delta_1 = [0 < \{x_0\} < \{x_1\} = \{x_2\} < \{x_3\} < 1] \]
   of dimension 3. Then, after the cut the simplex is of the form
   \[ \Delta_2 = [0 = \{x_2\} < \{x_0\} < \{x_1\} < \{x_3\} < 1] \]
   of dimension 3, and then
   \[ \Delta_3 = [0 < \{x_2\} < \{x_0\} < \{x_1\} < \{x_3\} < 1] \]
   of dimension 4.
The different cases of clocks reset described above demonstrate that, unlike the situation with a single clock, the dimension of the trail can increase, decrease or stay the same after clocks are reset: clocks with the same fractional part can be separated, resulting in an increase of the dimension, while clocks whose fractional parts become identical (namely, 0) contribute to a decrease of the dimension.

The trail of a path is similar to zones (\cite{1}) and the simplices to regions (\cite{2}). We extend these notions by considering the global time $t$ as one of the clock variables, where the general properties and treatment are the same since $t$ behaves as the regular clocks except that $t$ is never reset and it does not participate in the transition guards.

Since all clocks advance at the same rate it is easy to prove by induction on the event index that all points of each simplex of the trail are covered (we refer here to open simplices): all possible differences between the fractional parts of the clocks are maintained when passing from one simplex to the other.

A transition guard is a conjunction of constraints $x_i = n$, which defines a hyperplane, or a constraint of the form $x_i < n$ ($x_i > n$), which defines a half-space. The intersection of these domains is a convex set parallel (or orthogonal) to the axes and with faces defined by integer values. The intersection with the simplicial trail is a union of simplices, maybe of smaller dimension than that of the simplicial trail. The timestamp of an event, which is the orthogonal projection to the $t$-axis of the event domain is then the union of the projections of simplices. Moreover, the timestamp forms a continuous set of time. This follows from the fact that both the trail and the clocks constraints form convex sets, hence also their intersection (the event domain) and the projection of the event domain to the $t$-axis are convex. Thus, we have

**Proposition 4.** The timestamp of each event involving multiple clocks is either an integral point or an (open, closed or half-open) interval between points $m$ and $n$, $m < n$, $m \in \mathbb{N}_0$ and $n \in \mathbb{N} \cup \infty$.

### 4 The Augmented Region Automaton

Given a timed automaton $A$, one can construct a region automaton $\mathcal{R}(A)$ (see \cite{1}), which is a finite automaton in which time is abstracted, and such that $A$ and $\mathcal{R}(A)$ define the same untimed language. Each vertex in the graph representation records a location in $A$ and a simplex (region) that the clocks reach after a transition, as well as regions that refer to clock values beyond the maximal integer value $M$ that appears in the guards of the automaton. The labels on the edges represent the taken actions. $\mathcal{R}(A)$ is path-connected, containing only the regions that can be reached from the initial region.

Since time is represented by a global clock $t$ that dictates the rate in which all clocks of $A$ advance then we can add this global clock to the region automaton and our regions will be those described in the previous section. The problem is that $t$ never resets, resulting in infinitely-many regions. One way to overcome this obstacle is to ignore the integral part of $t$ and refer only to its fractional part. Hence, we can construct a finite region automaton that includes $t$, which
we call the augmented region automaton, denoted by $\mathcal{R}_t(A)$, and we will show that it captures the information needed for the timestamp of $A$.

In order to gain what we lost by ignoring the integral part of $t$, we write on each edge, in addition to the action label, a non-negative integral value, which equals the difference in the integral part of $t$ between the source and the target regions. Thus, if $t_0, t_1$ are some values of $t$ in the intervals of the source and target regions respectively then the time label on the transition edge will be $[t_1] - [t_0]$.

When a region is reached with the value $x_i > M$, $i > 0$, $M$ the maximal integer that appears in the transition guards of the automaton, then the value of $x_i$ in the target region is denoted by $\top$, with no fractional value (but we keep track of the integral parts and the order of the fractional parts of the clocks that did not pass $M$).

When a transition is made with all clocks passing $M$ then the delay of the transition is not bounded from above. Then we write on the time label of such an edge the minimal integral (non-negative) time difference $m \in \mathbb{N}_0$ between the source and the target region, and mark it with a star, $m^*$, to indicate that it represents the set

$$m^* = \{n \in \mathbb{N}_0 | n \geq m\}.$$  

The value of $m$ can be computed by $m = [t_1] - [t_0]$, where $t_0$ is a value of $t$ in the source region and $t_1$ is an interval value of $t$ at the time when all clocks pass the value $M$. In the target region the values of the clocks $x_i$, $i > 0$, are then either $\top$ or 0, with the latter being in case of a reset. Notice that a transition from an infinite region (all regular clocks have value $\top$) is always expressed by an out-going edge with time label $0^*$.

The augmented region automaton contains more information than the region automaton, but because of clock resets the exact transition guards may not be recovered. For example, let $2 < x < 3$ and $0 < \{t\} < \{x\} < 1$ and let the transition be with the reset of $x$ and on the guard $2 < x < 3$ or alternatively on the guard $x = 3$. Then the target region is the same in both cases, but the defined language may be different. Thus, if we are interested in the full information of the TA we may add the clock constraints in places where needed. However, when it comes to the timestamp of the automaton, the information in $\mathcal{R}_t(A)$ suffices because we are only interested in the question whether some augmented region can be reached and then we know that the whole region can be reached (when considering the union of of runs) and the transition guards may be ignored.

5 The Infinite Augmented Region Automaton

In the augmented region automaton $\mathcal{R}_t(A)$ each region refers only to the fractional part of the clock $t$ and since the number of regions remains finite we get a finite automaton. If we now consider also the time labels of the edges then we can reconstruct the integral values of $t$. But then we end up with infinitely-many regions and the resulted region automaton is infinite. However, we will show that this automaton, which we denote by $\mathcal{R}_t^\infty(A)$, has an almost periodic structure
and in the next section we will apply this property to construct a finite folding of $R^t_\infty(A)$.

The time line that is associated with $R^t_\infty(A)$ is discretized (quantized) into the alternating point and open unit interval. These compose the time regions of the clock $t$, namely: $\{0\}$, $(0,1)$, $\{1\}$, $(1,2)$, $\{2\}$, $(2,3)$,... Each such time region represents a time level, and the automaton is unfolded with respect to the clock $t$ so that each vertex belongs to a specific time level; hence, an unbounded region of $R^t(A)$ in which all regular clocks have value $\top$ is decomposed into infinitely-many regions with all regular clocks still having the $\top$ value, but the value of $t$ changes.

In order to construct $R^t_\infty(A)$, we start with the directed graph representation of the augmented region automaton $R^t(A)$. Then we delete the integral time difference labels of the edges and instead add to each region the integral part of $t$. So, each region $R$ of $R^t_\infty(A)$ is of the form

$$R = (q, n, \Delta),$$

where $q$ is a location of $A$, $\Delta$ is the simplex of the fractional parts of the clocks $t, x_1, \ldots, x_r$, and

$$n = (n_0, n_1, \ldots, n_r) \in \mathbb{N}_0 \times \{0, 1, \ldots, M, \top\}^r$$

are the integral parts of the clocks: $n_0 \in \mathbb{N}_0$ is the integral part of $t$, and $n_1, \ldots, n_r$ are the integral parts of $x_1, \ldots, x_r$, with $\top$ referring to the half-bounded interval $(M, \infty)$, where $M$ is the maximal integer appearing in a transition guard of $A$.

If an edge of $R^t(A)$ is labeled by $m^*$ then it is replaced in $R^t_\infty(A)$ by infinitely-many edges of the same action label. The first edge goes to some region whose discrete time is $\{m\}$, the second edge goes to the region of time $(m, m+1)$, the third to the region of time $(m+1, m+1)$, and so on (or the first edge goes to a region of time $(m, m+1)$, then to $\{m+1\}$, and so on). Each of the regular clocks $x_i$, $i > 0$, in all the target regions is either of value $\top$ or of integral and fractional value 0 (in case $x_i$ is reset).

We remark that as with the region automaton, if we want to have the full information embedded in the original automaton $A$ we may add to $R^t_\infty(A)$ the clock constraints of the transitions.

Example 2. In Fig. 5 we see a TA $A$ containing a transition to an unbounded region (a). The translation into the infinite augmented region automaton $R^t_\infty(A)$ is seen in (b). In each region we see on the left the original location of $A$ (encircled), and in the center are the integral values of $t$ and $x$ (top) and the simplex (bottom). Notice that when the value of $x$ is greater than $M = 0$ it is marked as $\top$ and its fractional part is ignored.

Example 3. Let us look at the more complex example of Fig. 7 (a). In order to make the analysis simpler we changed the guard on the transition from location 1 to location 2 to be simpler (Fig. 6 (a)), so that in the resulting infinite augmented...
Fig. 5. (a) $A \in TA$; (b) $R_{\infty}^t(A)$, the infinite augmented region automaton; (c) $R_{\text{per}}^t(A)$, an almost periodic augmented region automaton of $A$

region automaton $R_{\infty}^t(A)$ (Fig. 6(b)) we can clearly see two different cycles of period 6 (encircled in dotted lines) (the edges with label $c$ are only partly shown). We then added the original guard between locations 1 and 2 (Fig. 7(a)). In the additional part in the graph of $R_{\infty}^t(A)$ (Fig. 7(b)) we see two more cycles, one of period 11 and one of period 5. We can still use the period 6, but the existence of cycles of other lengths result in longer time until the graph stabilizes.

6 Almost Periodicity

In this section we address the main topic of this note: revealing the time-periodic property of the TA. In addition to demonstrating the existence of periods, we show how they can be computed. The periodic nature of the region automaton is evident from the cycles it contains. Nonetheless, when treating time as a continuous variable then it is not clear whether these periodic characteristics are maintained. Indeed, when examining single runs the behavior may look non-periodic even when repeating the same cycle. However, when we look at the set of runs as a whole then the periodic nature comes to life similarly to its existence in the discrete region automaton.
Fig. 6. (a) The simplified $A \in \mathbb{eNTA}$; (b) $\mathcal{N}_\infty^u(A)$ with period 6
Fig. 7. (a) The original $A \in \mathcal{eNTA}$; (b) The additional part of $\mathcal{R}_{\infty}(A)$ with cycles of lengths 11 and 5
6.1 Strongly Connected Components

A Strongly Connected Component (SCC) in a finite directed graph is a non-trivial subgraph which is maximal with respect to the property that each vertex of it can be reached from any other vertex of the SCC. The SCC structure of a directed graph forms a Directed Acyclic Graph (DAG). The edges of a connected directed graph with an initial vertex may be partitioned into three: 1) cycle edges - edges belonging to some SCC; 2) ‘hairs’ and ‘bridges’ - edges that can be reached from some SCC but do not belong to any SCC: a bridge lies on a path connecting two SCCs, while a hair lies on a dead-end path; 3) prefix edges - edges that can be reached from the initial vertex but not from any SCC.

Hereafter, a ‘path’ means a path that starts at the initial vertex, unless otherwise stated. Each prefix edge can be reached by only finitely-many paths, hence there exists a global bound, such that that any edge which lies in a position beyond that bound in any path cannot be a prefix edge, and this property characterizes prefix edges. A hair or a bridge edge does not share this property, but similar to a prefix edge it can appear only once in a single path. On the other hand, there is no bound on the number of times a cycle edge can occur in a single path.

We come now to the graph $\mathcal{R}(A)$. It is a finite connected directed graph with an initial vertex, augmented with time labels on the edges. Given a path $\gamma$ in $\mathcal{R}(A)$, its duration $d(\gamma) \in \mathbb{N}_0$ is the sum of the time labels on the edges along $\gamma$. When the time label of an edge $e$ is $m^\ast$ (unbounded time) then when computing the duration of a path through $e$ the value $m$ is used, but the result is written as $d(\gamma) = s^\ast \equiv \{ n \in \mathbb{N}_0 | n \geq s \}$.

We call a path in which no edge of unbounded time appears a bounded path, and otherwise - an unbounded path. Clearly, there is an integer $m$ such each edge on a bounded path of $\mathcal{R}(A)$ is not a prefix edge once the duration of the path passes $m$. As for the non-prefix edges, there is mostly no upper bound to the time at which they can be reached. The exception can happen as a result of Zenoness: all transitions in an entire SCC may occur at the same discrete time (the integral time is constant but the fractional time may increase). When this is the case, we call the SCC singular. Otherwise, it is called regular.

**Lemma 1.** There exists a computable positive integer, $t_{reg}$, such that if $\gamma$ is a bounded path of $\mathcal{R}_\infty(A)$ of duration $t_{reg}$ or more then it contains a vertex belonging to some regular SCC.

**Proof.** If there is a bound on the duration of all paths of $\mathcal{R}_\infty(A)$ then the claim holds trivially. Otherwise, a path of $\mathcal{R}_\infty(A)$ is induced by a path of $\mathcal{R}(A)$. To prove the claim by contradiction, we may remove from $\mathcal{R}(A)$ all edges that belong to regular SCCs. The only cycles in the remaining graph belong to singular SCCs. Since each singular SCC happens at a constant integral time, the whole SCC may be contracted to a single vertex without changing the duration of

---

2 It is also common to consider single vertices with trivial loops as maximal connected components.
paths going through it. The resulting graph is a finite DAG and the claim now follows immediately since we are looking only at bounded paths, that is, there is a bound on the duration of the edges. □

6.2 A Period of $\mathcal{R}^t(A)$

Given a vertex $v$ of $\mathcal{R}^t(A)$, let $\text{cycles}(v)$ be the semigroup of all cycles in $\mathcal{R}^t(A)$ which start and terminate at $v$, with the trivial loop representing the identity element. We define

$$l(v) = \min_{\gamma \in \text{cycles}(v), d(\gamma) > 0} d(\gamma),$$

where the minimum is defined to be 0 in case there is no cycle of $v$ of positive duration.

**Definition 3.** A period $L = L(\mathcal{R}^t(A))$ of $\mathcal{R}^t(A)$ is

$$L := \text{lcm}_{l(v) > 0} l(v),$$

where the least common multiple is computed over all the positive values $l(v)$ of all the vertices of $\mathcal{R}^t(A)$. $L$ is defined to be 1 in case $l(v)$ vanishes for all the vertices of $\mathcal{R}^t(A)$.

We remark that for our purpose it is not necessary to consider cycles of the smallest duration for each vertex $v$, nor do we try to find the smallest period $L$, although this is possible. In fact, for what follows, let us assume that $L$ is greater than $M$, the maximal constant of the transition guards (we may always choose such an $L$).

6.3 Almost Periodicity of $\mathcal{R}^\infty(A)$

**Proposition 5.** Let $n_0 \geq t_{\text{reg}}$ and let $e$ be an edge of $\mathcal{R}^\infty(A)$,

$$e := [u = (q, (n_0, n_1, \ldots, n_r), \Delta) \xrightarrow{a} (q', (n'_0, n'_1, \ldots, n'_r), \Delta') = u'].$$

Then $\mathcal{R}^\infty(A)$ contains an edge $e'$

$$e' := [v = (q, (n_0 + L, n_1, \ldots, n_r), \Delta) \xrightarrow{a} (q', (n'_0 + L, n'_1, \ldots, n'_r), \Delta') = v'].$$  \hspace{2cm} (8)

**Proof.** Let $\gamma$ be a path which ends in the vertex $u$. If $\gamma$ is bounded then since $n_0 \geq t_{\text{reg}}$ then, by Lemma 4, $\gamma$ contains a vertex $w$ which belongs to a regular SCC. Since $l(w) \mid L$ there exists a closed path $\delta$ that starts and terminates at $w$ and is of duration $L$. Hence, we can form the path $\gamma'$, which is the same as $\gamma$ except that it contains the additional cycle $\delta$. Then $\gamma'$ terminates in the vertex $v$, which is identical to $u$ except for the value of $t$ which is shifted by $L$.

Otherwise, $\gamma$ contains an unbounded edge $e''$, and we can form the path $\gamma'$ by increasing the delay of $e''$ by $L$. The result is that the value of $t$ at the target vertex of $e''$ is increased by $L$, whereas the values of the other clocks remain unchanged (they are either $\top$ or 0), and the claim follows. □
We have seen that each out-going edge of time $t_{reg}$ or later repeats itself every $L$ time units. This means that if we partition the discrete time line starting at $t_{reg}$ into discrete intervals $T_i$ of duration $L$ (each $T_i$ contains $2L$ elements) then the subgraph $G_i$ of $\mathcal{R}^\infty_{\infty}(A)$ that is induced by the edges that start at a time within $T_i$ is embedded in the subgraph $G_{i+1}$ that corresponds to $T_{i+1}$. We remark that although each of the subgraphs $G_i$ contains finitely-many vertices (the augmented regions), the number of out-going edges may be infinite. For the next theorem let us suppose that $\mathcal{R}^\infty_{\infty}(A)$ is indeed infinite.

**Theorem 1.** The infinite augmented region automaton $\mathcal{R}^\infty_{\infty}(A)$ is almost periodic in the sense that there exists an index $i_0$ such that for $i \geq i_0$ the subgraphs $G_i$ defined above are equivalent: $G_i$ is identical to $G_{i+1}$ up to an $L$-shift in time.

**Proof.** By the proof of Proposition [5] for every vertex $u$ of $G_i$ there is an equivalent vertex $v$ of $G_{i+1}$. This includes also vertices of $G_i$ which do not have out-going edges. For each $i$, let $\bar{V}(G_i)$ be the set of equivalent classes of the vertices $V(G_i)$ (modulo time $L$). Then the sets $\bar{V}(G_i)$ satisfy:

$$\bar{V}(G_1) \subseteq \bar{V}(G_2) \subseteq \bar{V}(G_3) \subseteq \cdots .$$

But $\bar{V}(G_i)$ is finite, hence this sequence eventually stabilizes. That is, there exists $i_0$ such that

$$\bar{V}(G_{i_0}) = \bar{V}(G_{i_0+1}) = \bar{V}(G_{i_0+2}) = \cdots.$$  

The set of out-going edges of $G_i$ is equivalent to the set of out-going edges of $G_{i+1}$, $i \geq i_0$, since $G_i$ and $G_{i+1}$ contain equivalent vertices and the guards on the out-going edges of $A$ do not refer to time $t$ but only to the other clocks. By induction, we conclude that the subgraphs $G_i$, $i \geq i_0$, are equivalent.

Let $\bar{G}_i$ be the equivalence class of the subgraph $G_i$ modulo time $L$. We have seen that the sequence $\bar{G}_i$ is non-decreasing and eventually becomes constant. In fact, the stabilization index $i_0$ is computable by the following lemma.

**Lemma 2.** if $\bar{G}_i = \bar{G}_{i+1} = \bar{G}_{i+2}$ then $\bar{G}_i = \bar{G}_{i+j}$ for every $j \geq 0$.

**Proof.** It suffices to consider the (equivalent classes of the) vertices of the subgraphs. We need to show that $\bar{V}(G_{i+2}) = \bar{V}(G_{i+3})$ and the rest is by induction. So, given a vertex $v \in V(G_{i+3})$, we need to find an equivalent vertex $v' \in V(G_{i+2})$. If $v$ is reached by an edge from a vertex in $G_{i+1}$ or in $G_{i+2}$ then an equivalent vertex $v' \in V(G_{i+2})$ is reached from a vertex in $G_i$ or in $G_{i+1}$ respectively. Otherwise, $v$ is reached by an edge $e$ from a vertex $w$ in $G_i$ or earlier, and the time difference $d$ between $w$ and $v$ is greater than $2L$. This implies that $e$ corresponds to an unbounded edge $e'$ of $\mathcal{R}(A)$. Since $L > M$ then $d - L > M$ and there is another edge $e''$ in $\mathcal{R}_{\infty}(A)$ that corresponds to $e$, which connects $w$ to a vertex $v''$ of $G_{i+2}$, and this vertex $v''$ is equivalent to $v$.

As we have seen, there exists an integral value of $t$, which we denote $t_{per}$, after which $\mathcal{R}^\infty_{\infty}(A)$ is periodic.
Theorem 2. Given \( A \in \text{eNTA} \), there exists an integral time \( t_{\text{per}} > 0 \) and an integer \( L > 0 \), such that if an \( a \)-event of \( A \) occurs at time \( t > t_{\text{per}} \) then, for every \( K \in L\mathbb{Z} \), if \( t' = t + K > t_{\text{per}} \) there exists an \( a \)-event of \( A \) at time \( t' \).

Proof. If \( A \) is bounded in time then the claim holds vacuously. So, suppose \( A \) is not bounded in time. By Theorem 1 the infinite augmented region automaton is almost periodic. That is, from some time \( t_{\text{per}} \) and for some integer \( L > 0 \), if \( n, n' > t_{\text{per}} \) and \( n' = n + K \), for some \( K \in L\mathbb{Z} \), then for any region of \( \mathcal{R}_\infty(A) \) of integral time \( n \) there is an identical region (modulo \( L \) in time) of integral time \( n' \). By Proposition 4 if an \( a \)-event occurs at some point in time in some region then at any other point in time of that region an \( a \)-event occurs, which concludes the proof.

More generally, the language of a non-deterministic timed automaton with silent transitions is almost periodic in the sense that all suffixes that occur after \( t_{\text{per}} \) are repeated every \( L \) time units in the observable timed traces (the repetitions are not supposed to occur in the same timed trace).

Theorem 3. The language of \( A \in \text{eNTA} \) is almost periodic: there exists time \( t_{\text{per}} > 0 \) and integer \( L > 0 \), such that if \( \sigma = (t_1, a_1), \ldots, (t_{r-1}, a_{r-1}), (t_r, a_r), (t_{r+1}, a_{r+1}), \ldots, (t_{r+m}, a_{r+m}) \) is (an observable) timed trace of \( \Sigma(A) \) with \( t_r > t_{\text{per}} \) then, for each \( K \in L\mathbb{Z} \), if \( t_r + K > t_{\text{per}} \) there exists (an observable) timed trace \( \sigma' \in \Sigma(A) \) such that \( \sigma' = (t'_1, a'_1), \ldots, (t'_s, a'_s), (t_r + K, a_r), (t_{r+1} + K, a_{r+1}), \ldots, (t_{r+m} + K, a_{r+m}) \).

Proof. If \( A \) is bounded in time then the claim holds vacuously. Otherwise, it follows from Theorem 2 and the fact that one can reach at time \( t_r + K \) the same region as of time \( t_r \).

\[ \Box \]

7 An Almost Periodic Augmented Region Automaton

In the augmented region automaton \( \mathcal{R}_t(A) \) the augmented regions contain only the fractional part of the time clock \( t \). In the infinite augmented region automaton \( \mathcal{R}_\infty(A) \) the augmented regions contain the fractional as well as the integral part of \( t \). Between these extremes we compute in this section a finite automaton: an almost periodic augmented region automaton \( \mathcal{R}_{t_{\text{per}}}(A) \).

If \( \mathcal{R}_\infty(A) \) is time-bounded then it is a finite graph and we may take \( \mathcal{R}_{t_{\text{per}}}(A) \) to be \( \mathcal{R}_\infty(A) \). So, let us assume that \( \mathcal{R}_\infty(A) \) is unbounded by time.

The construction of \( \mathcal{R}_{t_{\text{per}}}(A) \) is based on the almost periodicity structure of \( \mathcal{R}_\infty(A) \). As we have seen, we can compute an integral time, that we denoted by \( t_{\text{per}} \), after which \( \mathcal{R}_\infty(A) \) is periodic. Similarly, \( \mathcal{R}_{t_{\text{per}}}(A) \) is partitioned into a non-periodic subgraph and a periodic subgraph, with edges connecting the non-periodic part to the periodic one. The non-periodic subgraph consists of the
connected prefix subgraph of $R^t_\infty(A)$ that is induced by all vertices of integral time $t < t_{\text{per}}$.

The periodic subgraph of $R^t_{\text{per}}(A)$ is the projection of the subgraph $R^t_\infty(A)$ of integral time $|t| \geq t_{\text{per}}$, so that $R^t_{\text{per}}(A)$ is almost a covering space of the periodic part of $R^t_{\text{per}}(A)$. It is not precisely a covering space because of the unbounded edges. An unbounded edge of $R^t(A)$ (that is, an edge marked by $m^*$) is lifted to infinitely-many edges of $R^t_{\text{per}}(A)$, with infinitely-many of them of the same source vertex. These edges are then projected to finitely-many edges of $R^t_{\text{per}}(A)$, and the covering property is lost. Instead, each of the edges of $R^t_{\text{per}}(A)$ in the image is marked with a star ($*$) so that this mark indicates that for a single source vertex there are infinitely-many target vertices (in $R^t_\infty(A)$), all of the same time modulo $L$ (this is in addition to the fact that an (unmarked or marked) edge of the periodic part of $R^t_{\text{per}}(A)$ corresponds to infinitely-many edges of $R^t_\infty(A)$ with infinitely-many source vertices but with each edge having a single target vertex).

The fiber of a vertex of $R^t_{\text{per}}(A)$ of integral time $t_{\text{per}} \leq n < t_{\text{per}} + L$ consists of the vertices of integral time $n + kL$, $k = 0, 1, 2, \ldots$. Hence, we write the integral time of this vertex of $R^t_{\text{per}}(A)$ as $n + LN_0$, that is, the integral time at the periodic part of $R^t_{\text{per}}(A)$ is given modulo $L$, and we refer to such a vertex as a periodic vertex (or periodic region).

The duration of a bounded path with a single edge is always less than $L$. Thus, if the edge $e$ of $R^t_{\text{per}}(A)$ connects the vertex of integral time $n_1 + LN_0$ to the vertex of integral time $n_2 + LN_0$, $t_{\text{per}} \leq n_1, n_2 < t_{\text{per}} + L$, then the following holds. If $n_1 \leq n_2$ then $e$ represents the infinitely-many edges of $R^t_{\text{per}}(A)$ connecting the corresponding vertices of integral times: $n_1 \rightarrow n_2, n_1 + L \rightarrow n_2 + L, n_1 + 2L \rightarrow n_2 + 2L$, and so on, each edge in a different sheet. If $n_1 > n_2$ then the corresponding edges of $R^t_\infty(A)$ are: $n_1 \rightarrow n_2 + L, n_1 + L \rightarrow n_2 + 2L, n_1 + 2L \rightarrow n_2 + 3L$, and so on, each edge connecting a sheet to the one above it.

Example 4. The example shown in Fig. 8 is taken from [1] as an example of non-periodicity since the time difference between an $a$-event and the following $b$-event is strictly decreasing. However, the periodicity among the collection of timed traces is seen in the almost periodic augmented automaton, where the period here is of size 1, and the vertices in times $(2, 3) + \mathbb{N}_0$ and $3 + \mathbb{N}_0$ are periodic. Notice also that there are edges marked with ($*$) which represent infinitely-many edges with the same source.

8 The Timestamp

The timestamp of the timed automaton $A$, denoted $\mathcal{T}S(A)$, is the union of the timestamps of all observable events of $A$. By Theorem 2 and Proposition 4 we have the following.
Theorem 4. The timestamp of a timed automaton $A$ is an almost periodic subset of $\mathbb{R}_{\geq 0}$, which can be effectively computed. It consists of a union of integral points and open unit intervals with integral end-points.

Example 5. The timestamp of the example shown in Fig. 8 is for the $a$-events $\tau_S^a(A) = \mathbb{N}$, and for the $b$-events $\tau_S^b(A) = [1, \infty)$.

The timestamp is an abstraction of the language of the timed automaton. It does not preserve the timestamp of single timed traces. However, the timestamp is almost periodic and computable, hence the timestamp inclusion problem is decidable. Thus, due to the general undecidability of the language inclusion problem in non-deterministic timed automata, one can use timestamps for refutation purpose.

Proposition 6. Given two cNTA$\tilde{s}$ $A$ and $B$ then $\tau_S(A) \not\subseteq \tau_S(B)$ implies $\mathcal{L}(A) \not\subseteq \mathcal{L}(B)$. 

Fig. 8. a) $A \in TA$; b) $\mathcal{N}^r_{\text{per}}(A)$, an almost periodic augmented region automaton of $A$. 

**Theorem 4.** The timestamp of a timed automaton $A$ is an almost periodic subset of $\mathbb{R}_{\geq 0}$, which can be effectively computed. It consists of a union of integral points and open unit intervals with integral end-points.

**Example 5.** The timestamp of the example shown in Fig. 8 is for the $a$-events $\tau_S^a(A) = \mathbb{N}$, and for the $b$-events $\tau_S^b(A) = [1, \infty)$.

The timestamp is an abstraction of the language of the timed automaton. It does not preserve the timestamp of single timed traces. However, the timestamp is almost periodic and computable, hence the timestamp inclusion problem is decidable. Thus, due to the general undecidability of the language inclusion problem in non-deterministic timed automata, one can use timestamps for refutation purpose.

**Proposition 6.** Given two cNTA$\tilde{s}$ $A$ and $B$ then $\tau_S(A) \not\subseteq \tau_S(B)$ implies $\mathcal{L}(A) \not\subseteq \mathcal{L}(B)$.
Since the timestamp does not care about single traces, we can construct a simpler timed automaton with the same timestamp as the original automaton. For example, we may partition the timed automaton \( A \) into the ‘disjoint union’ of timed automata \( A_1, \ldots, A_r \) according to the observable event labels \( \Sigma = \{a_1, \ldots, a_r\} \). Each \( A_i \) contains observable events of type \( a_i \) only, for \( i = 1, \ldots, r \), after replacing all observable events of \( A \) of type \( a_j, j \neq i \), with silent events.

A further simplification is to construct a deterministic timed automaton with a single clock, which we call a *timestamp automaton* associated with the given timed automaton. Each \( A_i \) may be formed in the shape of a ‘balloon tied to a ribbon’. The ribbon part is the finite prefix, while the balloon represents the periodic part (if exists) of \( \mathcal{T} \mathcal{S}^{a_i}(A) \), where \( \mathcal{T} \mathcal{S}^{a_i}(A) \) is the \( a_i \)-part of the timestamp \( \mathcal{T} \mathcal{S}(A) \). When the periodic part of \( \mathcal{T} \mathcal{S}^{a_i}(A) \) contains an event that occurs at an integral time then the clock, which resets at the initial location, resets again when entering the location \( q \) where the cycle (the balloon part) starts and then once more when finishing the cycle and reaching \( q \) again. Each location of this timed automaton has exactly one out-going edge, with a guard matching the next time interval corresponding to the next \( a_i \)-interval in the timestamp \( \mathcal{T} \mathcal{S}^{a_i}(A) \) (see Fig. 9 (a)). When the periodic part does not include an event on an integral time then we let the clock reset on a transition before entering the cycle (thus, on any fractional time), and then the transitions within the periodic part can occur at integral times (see Fig. 9 (b)). Finally, we connect all the \( A_i \) by joining their initial locations. We just need to make sure that each ribbon is non-degenerate even if we can form the periodic part right from the initial state.

**Example 6.** Let \( A \) be a timed automaton with timestamp
\[
\mathcal{T} \mathcal{S}^{a_i}(A) = (1, 3] \cup \{5\} \cup (6 + ([0, 2] \cup \{3\} \cup (8, 18)) + 21\mathbb{N}_0),
\]
\[
\mathcal{T} \mathcal{S}^{b_i}(A) = [0, 1] \cup (2, 4) \cup \{5\} \cup (6 + ([0, 1] \cup (1, 2) \cup (5, 6) \cup (8, 9)) + 10\mathbb{N}_0),
\]
\[
\mathcal{T} \mathcal{S}^{c_i}(A) = [1, 4] \cup \{6\} \cup (10, \infty).
\]
In Fig. 9 a timestamp automaton of \( A \) is given.

**Example 7.** The language of the timed automaton \( A \in \text{eNTA} \) of Fig. 10 (a) is \( \Sigma(A) = \{(t_0, a), (t_1, a), \ldots, (t_n, a) \mid i < t_i < i + 1, i \in \mathbb{N}_0\} \). The timestamp of \( A \) is \( \mathcal{T} \mathcal{S}(A) = \{(n, n + 1) \mid n \in \mathbb{N}_0\} \). \( A \) is non-determinizable since each event occurs between the next pair of successive natural numbers, hence the guards need to be able to capture all natural numbers. But the constraints of the guards cannot refer to clocks reset on the \( a \)-events because these events occur on non-integral times. So, they have to refer to the only clock that can be reset at an integral time, and that is the clock that is reset at time 0 (and cannot be reset again at an integral time). However, since the automaton is finite, only a finite number of integers can appear in the guards of the transitions, making the mission of referring to all natural numbers impossible. Nevertheless, the timed automaton in Fig. 10 (b) is deterministic and has the same timestamp as that of \( A \), so it may serve as a timestamp automaton associated with \( A \).
8.1 First Timestamp

We give here a simple application of the timestamp construction. The universality problem asks whether the language of a TA consists of all possible timed traces. It is known \cite{1} to be undecidable in general. Here we address a simpler problem, which is decidable. Given a fixed positive integer $k$, we say that the language $L(A)$ has the $k$-bounded universality property if $L_k(A) := \{w \in L(A) \mid |w| \leq k\}$ contains all possible observable timed traces of length at most $k$.

**Proposition 7.** The 1-bounded universality problem for eNTA is decidable.

**Proof.** We add to $A$ a ‘sink’ location $q_{sink}$ and redirect each observable transition to it. Then we compute the timestamp of the connected component of the initial state and check whether it equals $\mathbb{R}_{\geq 0}$. □

In fact, the same construction for computing the first timestamp may be used for comparing $A_1, A_2 \in$ eNTA. Let $L = \text{lcm}(L_1, L_2)$, where $L_1, L_2$ are periods of $A_1, A_2$ respectively, and let $t_{per} = \max(t_{per_1}, t_{per_2})$. Then we compute the first timestamp of $A_1, A_2$ with period parameters $t_{per}$ and $L$. Due to the almost
periodicity of the timestamps it suffices to conduct the comparison up to time \( t_{\text{per}} + L \). We then have the following generalization of Proposition 7.

**Proposition 8.** The 1-bounded language inclusion problem for eNTA is decidable.

Notice that the set of timed traces of observable length 1 may be unbounded by length or by time since each such timed trace may contain a prefix of silent transitions, and these prefixes may be unbounded in length or time due to cycles of silent transitions, also with clock reset. Hence, the bounded-time \([13]\) and bounded-length \([14]\) techniques are not applicable here.

**References**

1. Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.
2. Rajeev Alur, Limor Fix, and Thomas A. Henzinger. Event-clock automata: A determinizable class of timed automata. *Theor. Comput. Sci.*, 211(1-2):253–273, 1999.
3. Rajeev Alur, Robert P. Kurshan, and Mahesh Viswanathan. Membership questions for timed and hybrid automata. In *Proceedings of the 19th IEEE Real-Time Systems Symposium, Madrid, Spain, December 2-4, 1998*, pages 254–263, 1998.
4. Rajeev Alur and P. Madhusudan. Decision problems for timed automata: A survey. In *Formal Methods for the Design of Real-Time Systems, International School on Formal Methods for the Design of Computer, Communication and Software Systems, SFM-RT 2004, Bertinoro, Italy, September 13-18, 2004, Revised Lectures*, pages 1–24, 2004.
5. Eugene Asarin and Oded Maler. As soon as possible: Time optimal control for timed automata. In *Hybrid Systems: Computation and Control, Second International Workshop, HSCC’99, Berg en Dal, The Netherlands, March 29-31, 1999, Proceedings*, pages 19–30, 1999.
6. Christel Baier, Nathalie Bertrand, Patricia Bouyer, and Thomas Brihaye. When are timed automata determinizable? In *ICALP (2)*, pages 43–54, 2009.
7. Béatrice Bérard, Antoine Petit, Volker Diekert, and Paul Gastin. Characterization of the expressive power of silent transitions in timed automata. *Fundam. Inform.*, 36(2-3):145–182, 1998.
8. Patricia Bouyer, Catherine Dufourd, Emmanuel Fleury, and Antoine Petit. Updatable timed automata. *Theor. Comput. Sci.*, 321(2-3):291–345, 2004.
9. Conrado Daws and Stavros Tripakis. Model checking of real-time reachability properties using abstractions. In *Proceedings of Tools and Algorithms for Construction and Analysis of Systems, TACAS ’98, Lisbon, Portugal*, pages 313–329, 1998.
10. Olivier Finkel. Undecidable problems about timed automata. In *FORMATS*, pages 187–199, 2006.
11. Joël Ouaknine, Alexander Rabinovich, and James Worrell. Time-bounded verification. In *CONCUR 2009 - Concurrency Theory, 20th International Conference, CONCUR 2009, Bologna, Italy, September 1-4, 2009. Proceedings*, pages 496–510, 2009.
12. Joël Ouaknine and James Worrell. On the language inclusion problem for timed automata: Closing a decidability gap. In *19th IEEE Symposium on Logic in Computer Science (LICS 2004), 14-17 July 2004, Turku, Finland, Proceedings*, pages 54–63, 2004.
13. Joël Ouaknine and James Worrell. Towards a theory of time-bounded verification. In *ICALP (2)*, pages 22–37, 2010.

14. Amnon Rosenmann, Florian Lorber, Dejan Nickovic, and Bernhard K. Aichernig. Bounded determinization of timed automata with silent transitions. Submitted.