Untimed Language Preservation in Timed Systems [⋆]

Ocan Sankur

LSV, CNRS & ENS Cachan, France
sankur@lsv.ens-cachan.fr

Abstract. Timed automata are a model that is extensively used in formal verification of real-time systems. However, their mathematical semantics is an idealization which assumes perfectly precise clocks, but does not correspond to real hardware. In fact, it is known that imprecisions, however small they may be, may yield extra behaviours. Several works concentrated on a relaxation of the semantics of timed automata to model the imprecisions of the clocks. Algorithms were given, first for safety, then for richer linear-time properties, to decide the *robustness* of timed systems, that is, the existence of a bound on the imprecisions under which the system satisfies a given property. In this work, we study a stronger notion of robustness: we show how to decide whether the untimed language of a timed automaton is preserved under small enough imprecisions, and provide a bound on the imprecision parameter.

1 Introduction

Timed automata [2] are a well established model in real-time systems design. These allow the modeling, model-checking and synthesis of systems with timing constraints, and several mature tools are today available. Implementing such mathematical models on physical machines is an important step in practical verification, and it is a challenging problem in timed systems. In fact, it is known that perturbations on clocks, however small they may be, can lead to a different semantics than the mathematical semantics in timed automata [15,11] (see Fig. 1 below). One way of modelling these perturbations is to *enlarge* all the guards, that is to transform any timing constraint of the form "$x \in [a, b]$" into "$x \in [a - \Delta, b + \Delta]$" for some parameter $\Delta > 0$. In fact, the resulting semantics is an overapproximation of a concrete implementation semantics, called the *program semantics* studied in [12], which corresponds to the execution of timed automata by a simple microprocessor with a digital clock. An important problem is then to determine whether a given timed automaton is *implementable*, that is, whether its implementation is correct with respect to a given property, in a fast enough hardware. For this purpose, *robust model-checking*, which asks for the existence of a parameter Δ under which the resulting enlarged semantics is

⋆ This work has been partly supported by projects DOTS (ANR-06-SETI-003) and ImpRo (ANR-10-BLAN-0317)

correct, was proven decidable for safety properties [15,11] and for richer linear-time properties [8,9]. If robust model-checking succeeds, then the given timed automaton is implementable in a fast enough hardware in the sense of [12].

In this work, we study a stronger notion of robustness which requires untimed language equivalence between the original and the enlarged automaton for some value of the parameter Δ. We call such timed automata *language robust.* In particular, if a timed automaton is language robust, then any (untimed) language based property (such as linear-time properties) proven for $[\![\mathcal{A}]\!]$ will be preserved in $[\![\mathcal{A}_\Delta]\!]$ for small enough Δ, hence also in the program semantics mentioned above. We show that language robustness is decidable in EXPSPACE for general timed automata, and we identify a class for which it can be decided in PSPACE. Note that the high complexity of our algorithm in the general case is not surprising, since deciding untimed language inclusion for timed automata is already EXPSPACE-complete [10]. In order to establish our results, we revisit the results of [11] and generalize these to a more general setting, taking into account the untimed languages (Section 4). Then, we prove a Ramsey-like combinatorial theorem on directed paths, which has an independent interest (Section 5). The proof of the main result (Section 6) combines these independent results.

Related work A closely related line of work considers clock drifts, where clocks can have different rates [15,12,11,3,13] (this is equivalent to enlargement under some assumptions [11]). A solution based on modelling the perturbations using timed automata was suggested in [1], but their approach suffer from the fact that the verification results obtained in some platform may not hold in a faster (or more precise) platform (this holds in our setting, see Subsection 2.3). Other notions of robustness have been investigated, mainly to *remove* “isolated” or “unlikely” behaviours using topological and probabilistic methods (*e.g.* [14,5]), but these do not make the link with physical implementations.

2 Preliminaries

2.1 Timed Automata

A *labelled timed transition system (LTTS)* is a tuple $(S, s_0, \Sigma, \rightarrow)$, where S is the set of *states*, $s_0 \in S$ the initial state, Σ a finite alphabet, and $\rightarrow \subseteq S \times (\Sigma \cup \mathbb{R}_{\geq 0}) \times S$ the *transition* relation.

Given a finite set of clocks $\mathcal{C}$, we call *valuations* the elements of $\mathbb{R}^{\mathcal{C}}_{\geq 0}$. For a subset $R \subseteq \mathcal{C}$ and a valuation v, we write $v[R \leftarrow 0]$ for the valuation defined by $v[R \leftarrow 0](x) = v(x)$ for $x \in \mathcal{C} \setminus R$ and $v[R \leftarrow 0](x) = 0$ for $x \in R$. Given $d \in \mathbb{R}_{\geq 0}$, the valuation $v + d$ is defined by $(v + d)(x) = v(x) + d$ for all $x \in \mathcal{C}$. We extend these operations to sets of valuations in the obvious way. For two valuations u and v, we let $d_\infty(u, v) = \max_{x \in \mathcal{C}} |u(x) - v(x)|$.

Given a clock set $\mathcal{C}$, a *guard* is a formula generated by the grammar $\Phi_{\mathcal{C}} ::= k \leq x \mid x \leq k \mid \Phi_{\mathcal{C}} \wedge \Phi_{\mathcal{C}}$, where k ranges over $\mathbb{Q}_{\geq 0}$ and x over $\mathcal{C}$. We define the *enlargement* of a guard by $\Delta \in \mathbb{Q}_{\geq 0}$ as $\langle k \leq x \rangle_\Delta = k - \Delta \leq x$, and

$\langle x \leq k \rangle_\Delta = x \leq k + \Delta$, for $x, y \in \mathcal{C}$ and $k \in \mathbb{Q}_{\geq 0}$. The enlargement of a guard g, denoted by $\langle g \rangle_\Delta$, is obtained by enlarging all atomic formulas.

A valuation v *satisfies* a guard g, denoted $v \models g$, if all formulas are satisfied when each $x \in \mathcal{C}$ is replaced by $v(x)$.

Definition 1. *A timed automaton $\mathcal{A}$ is a tuple $(\mathcal{L}, \mathcal{C}, \Sigma, l_0, E)$, consisting of finite sets $\mathcal{L}$ of* locations*, $\mathcal{C}$ of* clocks*, Σ of* labels*, $E \subseteq \mathcal{L} \times \Phi_{\mathcal{C}} \times \Sigma \times 2^{\mathcal{C}} \times \mathcal{L}$ of edges, and $l_0 \in \mathcal{L}$, the* initial location*. An edge $e = (l, g, \sigma, R, l')$ is also written as $l \xrightarrow{g,\sigma,R} l'$, where g is called the* guard *of e.*

A timed automaton $\mathcal{A}$ is *integral* if all constants that appear in its guards are integers. For any $\Delta \in \mathbb{Q}_{\geq 0}$, $\mathcal{A}_\Delta$ will denote the timed automaton where all guards are enlarged by Δ. We will refer to finite automata with no clocks as *untimed* automata.

Definition 2. *The semantics of a timed automaton $\mathcal{A} = (\mathcal{L}, l_0, \mathcal{C}, \Sigma, E)$ is an LTTS denoted by $[\![\mathcal{A}]\!]$, over alphabet Σ, whose state space is $\mathcal{L} \times \mathbb{R}_{\geq 0}^{\mathcal{C}}$. The initial state is $(l_0, \mathbf{0})$, where $\mathbf{0}$ denotes the valuation where all clocks have value 0. Transitions are defined as $(l, v) \xrightarrow{\tau} (l, v + \tau)$ for any state (l, v) and $\tau \geq 0$, $(l, v) \xrightarrow{\sigma} (l', v')$, for any edge $l \xrightarrow{g,\sigma,R} l'$ in $\mathcal{A}$ such that $v \models g$ and $v' = v[R \leftarrow 0]$.*

A *run* of $[\![\mathcal{A}]\!]$ is a finite or infinite sequence $\rho = (q_i, \tau_i, \sigma_i)_{i \geq 0}$ where q_i is a state of $[\![\mathcal{A}]\!]$, $\tau_i \in \mathbb{R}_{\geq 0}$ and $\sigma_i \in \Sigma$, such that for each $i \geq 0$, $q_i \xrightarrow{\tau_i} q_i' \xrightarrow{\sigma_i} q_{i+1}$ for some state q_i'. A run is *initialized* if $q_0 = (l_0, \mathbf{0})$. A *trace* is a word in $\Sigma^* \cup \Sigma^\omega$. We say that the above run ρ *follows the trace* $\sigma_0\sigma_1\ldots$, which is denoted by $\mathsf{tr}(\rho)$. We define $(\rho)_i = q_i$, the i-th state of ρ, and $\mathsf{first}(\rho) = q_0$ its first state. If ρ is finite, then we denote by $\mathsf{last}(\rho)$ its last state. We denote by $\rho_{i\ldots j}$ the run defined by ρ between states of indices i and j. Let $L([\![\mathcal{A}]\!])$ denote the set of (untimed) traces of the initialized finite and infinite runs of $[\![\mathcal{A}]\!]$. For any state q of $[\![\mathcal{A}]\!]$, $L([\![\mathcal{A}]\!], q)$ will denote the traces of the runs that start at q. The *length* of a finite run ρ is the length of its trace, and is denoted by $|\rho|$.

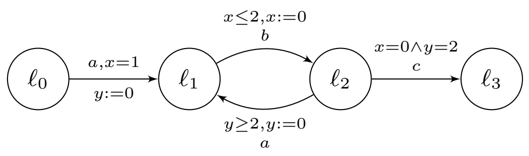


Fig. 1. The following is an example run of the above automaton: $(\ell_0, 0, 0) \xrightarrow{1} (\ell_1, 1, 1) \xrightarrow{a} (\ell_1, 1, 0) \xrightarrow{0.9} (\ell_1, 1.9, 0.9) \xrightarrow{b} (\ell_2, 0, 0.9) \xrightarrow{1.4} (\ell_2, 1.4, 2.3) \xrightarrow{a} (\ell_1, 1.4, 0)$. It can be seen that location ℓ_3 is not reachable in the runs of $[\![\mathcal{A}]\!]$. In fact, $L([\![\mathcal{A}]\!]) = (ab)^* + (ab)^*a + (ab)^\omega$. See [15].

We define the usual notion of *regions* and *region automaton* [2]. Consider an integral timed automaton $\mathcal{A}$ with clock set $\mathcal{C}$. Let K be the largest constant that appears in its guards. For any $(l,u),(l',v) \in \mathcal{L} \times \mathbb{R}^{\mathcal{C}}_{\geq 0}$, we let $(l,u) \simeq (l,v)$ if, and only if, $l = l'$ and

$$\begin{aligned}
&\text{either } \lfloor u(x) \rfloor = \lfloor v(x) \rfloor \text{ or } u(x), v(x) > K, && \forall x \in \mathcal{C},\\
&\text{and } \mathsf{frac}(u(x)) = 0 \iff \mathsf{frac}(v(x)) = 0, && \forall x \in \mathcal{C},\\
&\text{and } \mathsf{frac}(u(x)) < \mathsf{frac}(u(y)) \iff \mathsf{frac}(v(x)) < \mathsf{frac}(v(y)), && \forall x, y \in \mathcal{C},
\end{aligned}$$

where $\mathsf{frac}(\cdot)$ denotes the fractional part. The equivalence class of a state (l,v) for the relation $\simeq$ is denoted by $\mathsf{reg}((l,v)) = \{(l,u) \mid (l,u) \simeq (l,v)\}$, and called a *region* of $\mathcal{A}$. Note that our definition of regions includes locations. It is known that $\simeq$ has finite index [2]. For any region r, let $\overline{r}$ denote its topological closure. A region is *bounded* if it is a bounded subset of $\mathcal{L} \times \mathbb{R}^{\mathcal{C}}_{\geq 0}$. One can associate to $\mathcal{A}$, a finite (untimed) automaton $\mathcal{R}(\mathcal{A})$, called the *region automaton*, whose states are the regions of $\mathcal{A}$, and which has an edge of label $\sigma \in \Sigma$ from region r to r' whenever $q \xrightarrow{d} q'' \xrightarrow{\sigma} q'$ in $[\![\mathcal{A}]\!]$ for some $d \geq 0$ and $q \in r$ and $q' \in r'$. A *path* of $\mathcal{R}(\mathcal{A})$ is a sequence $\pi = (r_i, \sigma_i)_{i \geq 0}$ where r_i is a region and $\sigma_i \in \Sigma$, such that for each $i \geq 0$, $\mathcal{R}(\mathcal{A})$ has an edge from r_i to r_{i+1} with label σ_i. The i-th state of path π is denoted by $(\pi)_i$. Path π is *initialized* if $r_0 = \mathsf{reg}((l_0, \mathbf{0}))$. The *trace* of π is the word $\sigma_0\sigma_1\ldots$, which is denoted by $\mathsf{tr}(\pi)$. The set of traces of the finite or infinite paths of $\mathcal{R}(\mathcal{A})$ is denoted by $L(\mathcal{R}(\mathcal{A}))$. A finite path $\pi = (r_i, \sigma_i)_{0 \leq i \leq n}$ is a *cycle* if $r_0 = r_n$. A run $\rho = (q_i, \tau_i, \sigma_i)_{i \geq 0}$ of $[\![\mathcal{A}]\!]$ *follows a path* $(r_i, \sigma'_i)_{i \geq 0}$ of $\mathcal{R}(\mathcal{A})$ if $q_i \in \overline{r_i}$ and $\sigma_i = \sigma'_i$ for all $i \geq 0$. It is known that for any path π of $\mathcal{R}(\mathcal{A})$, there is a run of $[\![\mathcal{A}]\!]$ that follows π, starting from any state in $\mathsf{first}(\pi)$ and conversely. In particular, $L(\mathcal{R}(\mathcal{A})) = L([\![\mathcal{A}]\!])$ [2].

2.2 Restrictions on Timed Automata

Following [11,15], we defined timed automata with closed and *rectangular* guards (that is, we do not have *diagonal* constraints such as $k \leq x - y \leq l$). We also assume that all clocks are bounded above by some constant M. Considering closed guards is natural in our setting, since we are interested in the behaviour of the systems under positive enlargement. Assuming rectangular guards and bounded clocks is not restrictive in terms of expressiveness, but has an effect on the size of the models ([7]). As in [11,15,4], the only real restriction is the following. We consider timed automata where all clocks are reset at least once along any cycle of the region automaton; these are called *progress cycles*. A sufficient condition for a timed automaton to have only progress cycles is that any cycle of the underlying finite automaton resets all clocks at least once ([4]).

Although we prove our results for general timed automata with progress cycles, we also identify a subclass for which we improve the complexity of the problem we study. We call a timed automaton *concise* if its region automaton is deterministic (that is, from all states of the region automaton, there is at most one outgoing edge per label)[1].

[1] It would actually suffice to define conciseness by requiring that all states that satisfy the guards of edges with the same label to be language-equivalent (that the same

2.3 Robustness

By definition of the enlargement, we have $L([\![\mathcal{A}_\Delta]\!]) \subseteq L([\![\mathcal{A}_{\Delta'}]\!])$ for any $\Delta \leq \Delta'$, and in particular $L([\![\mathcal{A}]\!]) \subseteq L([\![\mathcal{A}_\Delta]\!])$ for any $\Delta \geq 0$. We are interested in the inverse inclusion, which does not always hold as we also noted in the introduction. In fact, if $\mathcal{A}$ denotes the timed automaton given in Figure 1, then for any $\Delta > 0$, all long enough words in $(ab)^*c$ belong to $L([\![\mathcal{A}_\Delta]\!])$ but not to $L([\![\mathcal{A}]\!])$ ([11]). (In fact location ℓ_3 is reachable in $[\![\mathcal{A}_\Delta]\!]$ iff $\Delta = 0$). Intuitively, this is due to the fact that at each cycle ab, the imprecisions can add up in $[\![\mathcal{A}_\Delta]\!]$, and the values of the clocks can deviate from what they can normally be in $[\![\mathcal{A}]\!]$.

Definition 3 (Language-robustness). *A timed automaton $\mathcal{A}$ is* language-robust *if there exists $\Delta > 0$ such that $L([\![\mathcal{A}]\!]) = L([\![\mathcal{A}_\Delta]\!])$.*

Informally, $\mathcal{A}$ is language-robust if $[\![\mathcal{A}_\Delta]\!]$ has no extra behaviour than $[\![\mathcal{A}]\!]$ for some $\Delta > 0$, in terms of untimed language. Observe that whenever $L([\![\mathcal{A}_\Delta]\!]) \subseteq L([\![\mathcal{A}]\!])$, we also have $L([\![\mathcal{A}_{\Delta'}]\!]) \subseteq L([\![\mathcal{A}]\!])$ for any $\Delta' < \Delta$. This is a desirable property, called "faster is better" [12,1], which means that once we prove the correctness of the system for some Δ, it remains correct on any faster platform.

3 Main Result

Theorem 1. *Let $\mathcal{A}$ be any timed automaton with progress cycles, and W the size of its region automaton. Let $K = W$ if $\mathcal{A}$ is concise, and $K = 2^W$ otherwise, and fix any $N_0 \geq 15 \cdot W \cdot |\mathcal{C}|^2 \cdot 2^{(|\mathcal{C}|+1)^2} \cdot (K+1)^2$. Then, there exists $\Delta > 0$ such that $L([\![\mathcal{A}_\Delta]\!]) = L([\![\mathcal{A}]\!])$ if and only if $L([\![\mathcal{A}_{\frac{1}{N_0}}]\!]) = L([\![\mathcal{A}]\!])$.*

Our main result, that is, the decidability of language-robustness is a direct corollary of the previous theorem. In fact, $\mathcal{A}_{\frac{1}{N_0}}$ can be transformed into a (language-)equivalent integral automaton by multiplying all constants by N_0. We will denote by $\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})$ the region automaton of the corresponding integral timed automaton. We can then check whether $\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})$ and $\mathcal{R}(\mathcal{A})$ recognize the same untimed language. We obtain the following complexity results.

Corollary 1. *For concise timed automata with progress cycles, language robustness can be decided in* PSPACE. *For general timed automata with progress cycles, language robustness can be decided in* EXPSPACE. [2]

Proof. Consider a concise timed automaton $\mathcal{A}$, and let $\mathcal{R}(\mathcal{A})$ denote its region automaton. Let $\mathcal{R}(\mathcal{A})^c$ denote the complement of $\mathcal{R}(\mathcal{A})$, which is not bigger than $\mathcal{R}(\mathcal{A})$ by hypothesis. Then, one can decide whether $L(\mathcal{A}_{\frac{1}{N_0}}) \cap L(\mathcal{R}(\mathcal{A})^c) \neq \emptyset$ in

untimed language is recognized). In fact, in this case, the region automaton can be made deterministic by leaving one (arbitrary) edge per label at each state.

[2] Note that in the original paper in MFCS 2011, the EXPSPACE result was stated in this corollary as 2EXPTIME, with an incorrect proof. The present proof is the corrected version.

polynomial space. In fact, the states of both $\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})$ and $\mathcal{R}(\mathcal{A})^c$ can be encoded in polynomial space (for $\frac{1}{N_0}$ given by the theorem for concise $\mathcal{A}$). Then, the usual non-deterministic procedure (*e.g.* [2]) that guesses an accepting path in the product of these can be carried out in polynomial space.

For general timed automata, we describe a non-deterministic exponential space algorithm to decide $L(\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})) \not\subseteq L(\mathcal{R}(\mathcal{A}))$. Observe that $\mathcal{R}(\mathcal{A})$ can be complemented using the subset construction, and that each state in the complemented automaton has exponential size (since there are exponentially many regions). Let us call the deterministic complement automaton $\mathcal{R}(\mathcal{A})^c$. The algorithm consists in guessing a path in $\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})$ while, in parallel, simulating the path in $\mathcal{R}(\mathcal{A})^c$. This can be done in exponential space since a state of $\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})$ can be represented in exponential space. The algorithm accepts if the simulating set becomes empty, and otherwise rejects after a doubly exponential number of steps. In fact, $L(\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})) \not\subseteq L(\mathcal{R}(\mathcal{A}))$ is equivalent to $L(\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})) \cap L(\mathcal{R}(\mathcal{A})^c) \neq \emptyset$, and in this case, the intersection contains a word of size at most doubly exponential, since the product automaton has this size. Consider now the general case. We give a non-deterministic exponential space algorithm to decide $L(\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})) \not\subseteq L(\mathcal{R}(\mathcal{A}))$. Observe that $\mathcal{R}(\mathcal{A})$ can be complemented using the subset construction, and that each state in the complemented automaton has exponential size (since there are exponentially many regions). Let us call the complement automaton $\mathcal{R}(\mathcal{A})^c$. The algorithm consists in guessing a path in $\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})$ while, in parallel, simulating the same path in the (deterministic) automaton $\mathcal{R}(\mathcal{A})^c$. This can be done in exponential space since a state of $\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})$ can be represented in exponential space. The algorithm accepts if the simulating set becomes empty, and otherwise rejects after a doubly exponential number of steps. In fact, $L(\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})) \not\subseteq L(\mathcal{R}(\mathcal{A}))$ is equivalent to $L(\mathcal{R}(\mathcal{A}_{\frac{1}{N_0}})) \cap L(\mathcal{R}(\mathcal{A})^c) \neq \emptyset$, but then this intersection contains a word of size at most doubly exponential, since the product automaton has this size. $\square$

In the rest of this paper, we present the proof of Theorem 1. We start with a study of the properties of enlarged timed automata (Section 4), and some combinatorial results (Section 5), then give the proof (Section 6).

4 Properties of the Semantics Under Enlargement

Let us fix a timed automaton $\mathcal{A}$ with $C > 0$ clocks. We start with the following result, which is a direct corollary of Lemma 3.14 we proved in [17]. It states that for any $\Delta > 0$, the trace of any run of $[\![\mathcal{A}_\Delta]\!]$ of length less then $O(\lfloor \frac{1}{\Delta} \rfloor)$ can be followed in $\mathcal{R}(\mathcal{A})$ too. An immediate implication is that if the length of the runs are fixed a priori, then a small enough enlargement has no effect on the behaviour of timed automata (in terms of untimed language). Figure 2 illustrates the construction of the following lemma.

Lemma 1 ([17]). *Fix any $n \in \mathbb{N}$ and $\Delta > 0$ such that $\Delta \leq \frac{1}{5nC^2}$. Let ρ be any run of $[\![\mathcal{A}_\Delta]\!]$. Then, for all $1 \leq i_0 \leq |\rho|$, there exists a region, denoted* *by $H(\rho, i_0, n)$, included in $\overline{\mathsf{reg}((\rho)_{i_0})}$, such that for all regions $r \subseteq \overline{H(\rho, i_0, n)}$, there is a path π of $\mathcal{R}(\mathcal{A})$ over the trace $\mathsf{tr}(\rho_{i_0 \dots \min(i_0+n, |\rho|)})$, with $(\pi)_1 = r$ and $\overline{(\pi)_j} \cap \overline{H(\rho, i_0 + j - 1, n)} \neq \emptyset$ for all $1 \leq j \leq |\pi|$.*

We are now interested in "long" or infinite runs. In [11], it is shown that for some timed automaton $\mathcal{A}$ (*e.g.* the one in Fig. 1) some regions that are not reachable in $[\![\mathcal{A}]\!]$ become entirely reachable in $[\![\mathcal{A}_\Delta]\!]$, for any $\Delta > 0$ (See also [15] for a similar analysis under *clock drifts*). An analysis of the behaviour of $[\![\mathcal{A}_\Delta]\!]$ shows that this is due to the accumulation of the "error" of Δ along some cycles of the region automaton. They give a characterization of those cycles of $\mathcal{R}(\mathcal{A})$ which cause this behaviour and get a decision procedure for safety properties. In this section, we revisit the analysis of the cycles of $\mathcal{R}(\mathcal{A})$ under enlargement, and prove the same results in a slightly more general setting. Roughly, we show that, the states that are reachable in $[\![\mathcal{A}_\Delta]\!]$ by repeating a single cycle are also reachable by repeating particular sets of cycles in any order. Our proofs follow [11].

A state whose valuation has integer components is called a *vertex*. For any region r, we denote by $V(r)$ the set of vertices of $\overline{r}$. Given a cycle π from a region r, we define the relation $\nu(\pi) \subseteq V(r) \times V(r)$, the *vertex map of* π, where $(q, q') \in \nu(\pi)$ if and only if there is a run in $[\![\mathcal{A}]\!]$, from q to q' following π. Note that for all $q \in V(r)$, there exists at least one $q' \in V(r)$ such that $(q, q') \in \nu(\pi)$, and $q'' \in V(r)$ such that $(q'', q) \in \nu(\pi)$ by the well-known properties of regions ([11]). Two cycles π_1 and π_2 are *equivalent* if $\mathsf{first}(\pi_1) = \mathsf{first}(\pi_2)$ and $\nu(\pi_1) = \nu(\pi_2)$. Let π be a cycle of $\mathcal{R}(\mathcal{A})$ starting at some region r. For any $k > 0$, we let $V_{\pi,k} = \{q \in V(r) \mid (q, q) \in \nu(\pi)^k\}$. This is the set of vertices $q \in \overline{\mathsf{first}(\pi)}$ for which there are runs following π^k that start and end at q. We define the convex hull of the union of these sets as $L_\pi = \mathsf{convex\text{-}hull}(\bigcup_{k>0} V_{\pi,k})$. It is clear from the definition of L_π that $L_\pi = L_{\pi'}$ for any equivalent cycles π and π'. The main result of this section is the following lemma, which generalizes Theorem 23 in [11] (see also Lemma 7.10 in [15]).

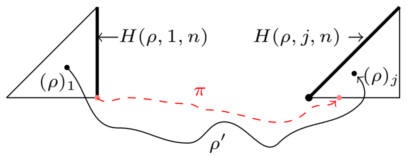


Fig. 2. A run ρ of $[\![\mathcal{A}_\Delta]\!]$. The leftmost triangle represents $\mathsf{reg}((\rho)_1)$, and the rightmost one $\mathsf{reg}((\rho)_j)$ (their corners, edges and interiors are subregions). By Lemma 1, there is a region $H(\rho, 1, n)$ such that starting from any region in $\overline{H(\rho, 1, n)}$, one can construct a path π of length n (the red dashed curve) such that $\overline{(\pi)_j}$ intersects $\overline{H(\rho, j, n)}$ for all j.

Lemma 2. *Let $\pi_1, \dots, \pi_p$ be equivalent cycles of $\mathcal{R}(\mathcal{A})$ that start in region r, and consider any $\Delta > 0$. Then, there exists $k > 0$ such that for any $q, q' \in \overline{r}$, and any word $w \in \{\pi_1, \dots, \pi_p\}^k$ there is a run in $[\![\mathcal{A}_\Delta]\!]$ from q to q' on word w.*

To prove this lemma, we first show that L_{π_1} is backward and forward reachable in $[\![\mathcal{A}]\!]$, from any point of $\overline{r}$, by iterating at least C times any of the equivalent cycles in any order (Lemma 5), and that any pair of points in L_{π_1} can be connected by a run of $[\![\mathcal{A}_\Delta]\!]$, again by iterating these cycles (Lemma 7).

A natural property of runs of timed automata is that convex combinations of two runs yield a run over the same word, as shown in the following lemma. We denote $\lambda(l, v) = (l, \lambda v)$ where $\lambda \in \mathbb{R}_{\geq 0}$, v is a valuation and l a location.

Lemma 3 ([11, Lemma 24], and [15, Lemma 7.1]). *Let π be a path in $\mathcal{R}(\mathcal{A})$, and let ρ and ρ' be runs in $[\![\mathcal{A}]\!]$ that follow π. Then for all $\lambda \in [0, 1]$, there exists a run ρ'' of $[\![\mathcal{A}]\!]$ following π, such that $(\rho'')_i = \lambda(\rho)_i + (1 - \lambda)(\rho')_i$ for all $1 \leq i \leq n$.* □

The following proposition provides a bound on the number of vertices of regions. It also implies that from each region, there is a finite number of cycles with pairwise distinct vertex maps. Remember that all clocks are bounded above by some constant, so we only need to consider bounded regions.

Lemma 4 ([11, Lemma 14]). *Any bounded region has at most $C+1$ vertices. Any point $u \in \mathbb{R}^C_{\geq 0}$ is a convex combination of the vertices of $\mathsf{reg}(u)$.* □

The following lemma states that, L_π is backward and forward reachable from any state of $\overline{\mathsf{first}(\pi)}$, by repeating at least C times cycles equivalent to π.

Lemma 5. *Let $\pi_1, \ldots, \pi_p$ be equivalent cycles of $\mathcal{R}(\mathcal{A})$, that all start in region state r, and fix any $q \in \overline{r}$. Then, for any $k \geq C$ and any path $w \in \{\pi_1, \ldots, \pi_p\}^k$, there exists $q_1, q_2 \in L_{\pi_1}$ and runs ρ_1 and ρ_2 of $[\![\mathcal{A}]\!]$ that follow w, such that $\mathsf{first}(\rho_1) = q$ and $\mathsf{last}(\rho_1) = q_1$; and $\mathsf{first}(\rho_2) = q_2$ and $\mathsf{last}(\rho_2) = q$.*

Proof. We first prove the statement when q is a vertex. As we already noted above, for all $v \in V(r)$, there exists at least one $v' \in V(r)$ such that $(v, v') \in \nu(\pi_1)$, and the number of vertices is at most $C+1$ by Lemma 4, so by repeating C times any cycles among $\pi_1, \ldots, \pi_p$, we get a sequence of vertices $v_1, \ldots, v_{C+1}$ such that $(v_i, v_{i+1}) \in \nu(\pi_1)$. But then, $v_i = v_j$ for some $i < j$, thus we have $v_i, v_{i+1}, \ldots, v_j \in L_{\pi_1}$. Now, we can extend the sequence $v_1 \ldots v_j$ to k vertices by repeating the cycle $v_i v_{i+1} \ldots v_j$. Clearly, this run can be constructed following any path w since π_i's are equivalent.

Consider now an arbitrary point q in $\overline{r}$. By Lemma 4, q can be written as a convex combination of the vertices of r. Let $v_1, \ldots, v_m$ denote the vertices of r, and $\lambda_1, \ldots, \lambda_m \geq 0$ be such that $\lambda_1 + \ldots + \lambda_m = 1$ and $q = \lambda_1 v_1 + \ldots + \lambda_m v_m$. As we showed above, for any v_i, there is a run in $[\![\mathcal{A}]\!]$ that follows w, from v_i to some vertex $v'_i \in L_{\pi_1}$. Lemma 3 yields the desired run. □

Lemma 6. *Let $\pi_1, \ldots, \pi_p$ be equivalent cycles in $\mathcal{R}(\mathcal{A})$. Then there exists $m > 0$ such that for all paths $w \in \{\pi_1, \ldots, \pi_p\}^m$, and for all $q \in L_{\pi_1}$, there is a run ρ in $[\![\mathcal{A}]\!]$ from q to q, following w.*

Proof. By definition of L_{π_1}, any $z \in L_{\pi_1}$ is a convex combination of a set of vertices v_i in L_{π_1}. But, for any vertex $v_i \in L_{\pi_1}$, there exists $m_i > 0$ such that $(v_i, v_i) \in \nu(\pi_1)^{m_i}$. So, there exists $m > 0$ such that $(v_i, v_i) \in \nu(\pi_1)^m$ for all $1 \leq i \leq k$. Now, the convex combination of these runs yield the desired run from q to q, by Lemma 3. □

The following lemma shows that any pair of states in L_π can be connected by a run in $[\![\mathcal{A}_\Delta]\!]$.

Lemma 7 ([11, Lemma 29]). *Let π be a cycle of $\mathcal{R}(\mathcal{A})$ that starts in region r and let $q \in L_\pi \cap \overline{r}$. Then for any $\Delta > 0$, and any $q' \in \overline{r}$ such that $d_\infty(q, q') \leq \frac{\Delta}{2}$, there is a run, in $[\![\mathcal{A}_\Delta]\!]$, from q to q' following π.* □

Proof (of Lemma 2). By Lemma 5, repeating at least C times the cycles $\pi_1, \ldots, \pi_p$ suffices to reach some point $q_1 \in L_{\pi_1}$. The same lemma provides a point $q_2 \in L_{\pi_1}$ which is backward reachable from q' by repeating C times any of these cycles. It follows from the definition of regions that for any pair of points q_1, q_2 that belong to a same region, one can find points $q_1 = u_0, u_1, \ldots, u_N = q_2$ in $\overline{r}$ where $N = \lceil \frac{1}{\Delta} \rceil$ such that $d_\infty(u_i, u_{i+1}) \leq \frac{\Delta}{2}$ for all $0 \leq i \leq N-1$. Let $m > 0$ be the bound provided by Lemma 6. Now, q_2 can be reached from q_1, by a run over any word $\{\pi_1, \ldots, \pi_p\}^{mN}$ by Lemma 7 (applied N times to pairs (u_i, u_{i+1})). □

5 Some Combinatorial Tools

In this section, we prove a Ramsey-like theorem for colored directed paths, which gives a lower bound on the length of monochromatic subpaths contained in these (Subsection 5.1). This improves, by an exponential, the result provided by a direct application of Ramsey's theorem [16]. In Subsection 5.2, we give a simple property on untimed finite automata accepting *ultimately universal* languages.

5.1 A Ramsey-like Theorem for Directed Paths

A *directed graph* is a pair $G = (V, E)$ where V is a finite set of *nodes* and $E \subseteq V \times V$, is the set of *edges*. A graph G is complete if for all $i, j \in V$ either $(i, j) \in E$ or $(j, i) \in E$. A graph is a *linearly-ordered complete graph*, if for some (strict) linear order $\prec$ on V, $(i, j) \in E$ iff $i \prec j$. The *degree* of a node v is $d(v) = |\{u \mid (v, u) \in E \text{ or } (u, v) \in E\}|$. Two nodes u and v are *connected* in the graph G if there exists a sequence $u = s_0, s_1, \ldots, s_k = v$ of nodes such that $(s_i, s_{i+1}) \in E$ or $(s_{i+1}, s_i) \in E$ for all $0 \leq i \leq k-1$. A graph is connected if all its nodes are connected. A *subgraph* of $G = (V, E)$ is a graph (V', E') such that $V' \subseteq V$ and $E' \subseteq E$. A *connected component* is a maximal connected subgraph. A *directed path* of G is a sequence of nodes $v_1, \ldots, v_n$ such that $(v_i, v_{i+1}) \in E$ for all $1 \leq i \leq n-1$, and its *length* is n. A *r-coloring* of a graph $G = (V, E)$ is a function $E \to \{1, \ldots, r\}$ that associates to each edge a color from $\{1, \ldots, r\}$. A path is *monochromatic* if all its edges are assigned the same color.

Our result is based on the following theorem from [6].

Theorem 2 ([6]). *Let G be a connected directed graph over n nodes such that, for some h, every node v satisfies $d(v) \geq h$. Then G contains a directed path of length* $\min(n, h+1)$. □

The main result of this subsection is the following theorem.

Theorem 3. *Let $G = (V, E)$ be a linearly-ordered complete graph over n nodes given with an r-coloring of its edges. Then G contains a monochromatic directed path of length $\lfloor\sqrt{n/r-2}\rfloor - 1$.*

Proof. Fix $h = \lfloor\sqrt{n/r-2}\rfloor - 1$. For each $1 \leq i \leq r$, define subgraph G_i which contains exactly the edges colored by i. Then, for each G_i, define G'_i by removing any node v (and any edge containing v) whose degree in G_i is less than h. In G'_i, any node has either none or at least h edges of color i. Let G' be the union of all G'_i's. To define G', we remove at most $(h-1)rn$ edges ($h-1$ edges per color and node). Thus, G' has at least $\binom{n}{2} - (h-1)rn$ edges. Then, one of the G'_i's has at least $\binom{n}{2}/r - (h-1)n$ edges, say G'_{i_0}. We show that G'_{i_0} has a connected component C with at least $\sqrt{(n-1)/r - 2(h-1)}$ nodes. In fact, consider a graph with N nodes and M edges. Let K denote the number of nodes of the largest connected component. Since there are at most N connected components, each of them containing at most $\binom{K}{2}$ edges, we get that $M \leq N\binom{K}{2}$, which implies $\sqrt{2M/N} \leq K$. Applying this to G'_{i_0}, we get the desired connected component C. Now, by construction of G'_{i_0}, all nodes of C have degree at least h in C (in fact, by maximality, all edges of G'_{i_0} adjacent to the nodes of C are also included in C). Then, by Theorem 2, C contains a directed path of length at least $\min(\sqrt{(n-1)/r - 2(h-1)}, h)$. But the first term of the min is greater than or equal to h by the choice of h. So, there is a directed path of length at least $\lfloor\sqrt{n/r-2}\rfloor - 1$ in G'_{i_0}, and this is a monochromatic path in G. $\square$

5.2 Ultimately Universal Languages

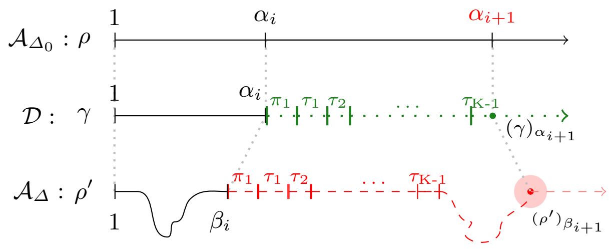
Let Σ denote a finite alphabet. We call a regular language $L \subseteq \Sigma^*$ *ultimately universal* if there exists $k \geq 0$ such that $\bigcup_{l \geq k} \Sigma^l \subseteq L$.

Lemma 8. *Let L be an ultimately universal language recognized by a deterministic untimed automaton with n locations. Then $\bigcup_{l \geq n-1} \Sigma^l \subseteq L$.*

6 Proof of the theorem

Fix any timed automaton $\mathcal{A}$ with $C \geq 1$ clocks. Let W denote the number of regions of $\mathcal{A}$. Let $\mathcal{D}$ denote a deterministic untimed automaton such that $L(\mathcal{D}) = L(\mathcal{R}(\mathcal{A}))$ (say, obtained by minimization), and let K denote the size of $\mathcal{D}$, which is at most 2^W ($K \leq W$ if $\mathcal{A}$ is concise). We let $M = 2^{(C+1)^2}\left((K+1)^2+2\right)$, $n \geq WM$ and fix any $\Delta_0 \leq \frac{1}{5nC^2}$. The theorem states that if $L(\llbracket\mathcal{A}\rrbracket) = L(\llbracket\mathcal{A}_\Delta\rrbracket)$ for some $\Delta > 0$, then $L(\llbracket\mathcal{A}\rrbracket) = L(\llbracket\mathcal{A}_{\Delta_0}\rrbracket)$. The only interesting case is when $\Delta < \Delta_0$ since otherwise $L(\llbracket\mathcal{A}\rrbracket) \subseteq L(\llbracket\mathcal{A}_{\Delta_0}\rrbracket) \subseteq L(\llbracket\mathcal{A}_\Delta\rrbracket)$ and the theorem follows. So let us suppose that $L(\llbracket\mathcal{A}\rrbracket) = L(\llbracket\mathcal{A}_\Delta\rrbracket)$ for some $\Delta < \Delta_0$. Let ρ be a run of $\llbracket\mathcal{A}_{\Delta_0}\rrbracket$. We will show that $\mathsf{tr}(\rho) \in L(\mathcal{D}) = L(\llbracket\mathcal{A}\rrbracket)$, which will prove the theorem.

Lemma 9. *Any path π of $\mathcal{R}(\mathcal{A})$ of length at least n can be factorized as $\pi = \pi_1\tau_1\tau_2\ldots\tau_{K-1}\pi_2$ where π_1, π_2 are paths, and τ_i's are equivalent cycles.*



$\mathcal{A}_{\Delta_0} : \rho$
1
α_i
α_{i+1}
$\mathcal{D} : \gamma$
1
α_i
π_1 τ_1 τ_2
τ_{K-1}
$(\gamma)_{\alpha_{i+1}}$
$\mathcal{A}_\Delta : \rho'$
1
β_i
π_1 τ_1 τ_2
τ_{K-1}
$(\rho')_{\beta_{i+1}}$

Fig. 3. An induction step in the proof of Theorem 1. First, ρ' is extended following path $\pi_1\tau_1\tau_2\ldots$ (shown in red dashed line), and for any long enough repetition of cycles $\tau_1, \ldots, \tau_K$, $\overline{H(\rho, \alpha_{i+1}, n)}$ (shown in pink filled circle) can be reached. Then, γ is extended to $(\gamma)_{\alpha_{i+1}}$ (shown in green loosely dotted line).

Proof. Since $n \geq W \cdot M$, by the Pigeon-hole principle, π contains a factor $t = t_1 \ldots t_M$ such that $\mathsf{first}(t_1) = \mathsf{first}(t_j)$ for all j. We apply Theorem 3 to get a further factorization of t. Consider a directed graph of the usual linear order $<$ over $\{1, \ldots, M\}$. To each edge (j, k) of the graph, where $j < k$, we assign as *color*, the vertex map $\nu(t_j t_{j+1} \ldots t_k)$. The number of colors is then bounded by $2^{(C+1)^2}$. Applying Theorem 3, we get that t contains a factor $\tau_1 \ldots \tau_{K-1}$, where $\tau_1 = t_{j_1} t_{j_1+1} \ldots t_{j_2}, \tau_2 = t_{j_2} t_{j_2+1} \ldots t_{j_3}, \ldots, \tau_{K-1} = t_{j_{K-1}} \ldots t_{j_K}$, for some $j_1 < j_2 < \ldots < j_K$, such that $\nu(\tau_1) = \nu(\tau_j)$ for all $1 \leq j \leq K$. □

Lemma 10. *Let $\pi = \pi_1\tau_1\tau_2 \ldots \tau_{K-1}\pi_2$ be a path of $\mathcal{R}(\mathcal{A})$ where π_1 and π_2 are paths and τ_i's are equivalent cycles. Then, there exists $k_0 > 0$ such that for all $q \in \overline{\mathsf{first}(\pi)}$, $q' \in \overline{\mathsf{last}(\tau_{K-1})}$, $k \geq k_0$, and any word $w \in \mathsf{tr}(\pi_1) \cdot (\mathsf{tr}(\tau_1) + \ldots + \mathsf{tr}(\tau_{K-1}))^k$, there is a run ρ' of $[\![\mathcal{A}_\Delta]\!]$ over w with $\mathsf{first}(\rho) = q$ and $\mathsf{last}(\rho) = q'$.*

The previous lemma follows from Lemma 7. We are now ready to prove our main theorem. The reader may follow the proof in Figure 3.

Proof (of Theorem 1). Consider ρ and the constants as defined above, and notice that $\Delta_0 \leq \frac{1}{N_0}$. Let $H(\rho, i, n)$ be the regions given by Lemma 1 for all $i \geq 0$. We will inductively construct the desired run γ of $\mathcal{D}$ with $\mathsf{tr}(\gamma) = \mathsf{tr}(\rho)$. At step i of the induction, we will define $\gamma_{\alpha_i \ldots \alpha_{i+1}}$ for some increasing sequence $(\alpha_i)_{i \geq 0}$ with $\alpha_0 = 1$. When constructing γ, we will also construct an auxiliary run ρ' of $[\![\mathcal{A}_\Delta]\!]$ in parallel, defining $\rho'_{\beta_i \ldots \beta_{i+1}}$ at each step i, for some increasing sequence $(\beta_i)_{i \geq 0}$ with $\beta_0 = 1$, and ensuring that $(\rho')_{\beta_i} \in \overline{H(\rho, \alpha_i, n)}$ and $L([\![\mathcal{A}_\Delta]\!], (\rho')_{\beta_i}) \subseteq L(\mathcal{D}, (\gamma)_{\alpha_i})$ for all $i \geq 0$.

– For $i = 0$, since ρ is an initialized run, we have $(\rho)_1 = (l_0, \mathbf{0})$ so $H(\rho, 1, n) = \mathsf{reg}(()l_0, \mathbf{0})$. We have $\alpha_0 = \beta_0 = 1$, $(\rho')_1 = (l_0, \mathbf{0})$ and $(\gamma)_1$ is the initial state of $\mathcal{D}$. We have $L([\![\mathcal{A}_\Delta]\!], \rho'_1) \subseteq L(\mathcal{D}, \gamma_1)$ by hypothesis.

– For any $i \geq 1$, suppose by induction that γ is defined between indices 1 and α_i and that $\rho'_{\beta_i} \in \overline{H(\rho, \alpha_i, n)}$. We will choose $\alpha_{i+1} > \alpha_i$ and $\beta_{i+1} > \beta_i$, and first define $\rho'_{\beta_i \ldots \beta_{i+1}}$ such that $(\rho')_{\beta_{i+1}} \in \overline{H(\rho, \alpha_{i+1}, n)}$, then define $\gamma_{\alpha_i \ldots \alpha_{i+1}}$. Let π be the path of $\mathcal{R}(\mathcal{A})$ which starts at $\mathsf{reg}((\rho')_{\beta_i})$, given by Lemma 1 for the run

$\rho_{\alpha_i \ldots |\rho|}$. If ρ is finite and $|\rho| - \alpha_i \leq n$, then $\mathcal{D}$ has a run from γ_{α_i} on word $\mathsf{tr}(\pi)$ (since $L(\llbracket \mathcal{A}_\Delta \rrbracket, \rho'_{\alpha_i}) \subseteq L(\mathcal{D}, \gamma_{\alpha_i})$) and we are done. Suppose now that ρ is either infinite, or $|\rho| - \alpha_i > n$. Then $|\pi| = n$, and by Lemma 9, π can be decomposed into $\pi = \pi_1 \tau_1 \ldots \tau_{K-1} \pi_2$ where τ_i's are equivalent cycles. We let $\alpha_{i+1} > \alpha_i$ such that $\mathsf{last}(\tau_{K-1})$ is the $(\alpha_{i+1} - \alpha_i)$-th state of π. $\llbracket \mathcal{A}_\Delta \rrbracket$ has a run from $(\rho')_{\beta_i}$ to some $z \in \mathsf{first}(\tau_1)$ following π_1 (in fact, $(\rho')_{\beta_i} \in \mathsf{first}(\pi_1)$). By construction of π, there exists $z' \in \overline{H(\rho, \alpha_{i+1}, n)} \cap \overline{\mathsf{last}(\tau_{K-1})} \neq \emptyset$, and by Lemma 10, for any $k \geq k_0$ and any word $w \in (\mathsf{tr}(\tau_1) + \ldots + \mathsf{tr}(\tau_{K-1}))^k$, there is a run, in $\llbracket \mathcal{A}_\Delta \rrbracket$, from z to z' following trace w. Let $\rho''(w)$ denote the run thus constructed from $(\rho')_{\beta_i}$ to z' on $\mathsf{tr}(\pi_1) \cdot w$. We let β_{i+1} s.t. $\rho'_{\beta_i \ldots \beta_{i+1}}(w) = \rho''(w)$ for an arbitrary w.

Now, $\mathcal{D}$ has a run from $(\gamma)_{\alpha_i}$ to some state q_0 over trace $\mathsf{tr}(\pi_1)$ because $L(\llbracket \mathcal{A}_\Delta \rrbracket, (\rho')_{\beta_i}) \subseteq L(\mathcal{D}, (\gamma)_{\alpha_i})$. Let $\mathcal{D}'$ denote the finite untimed automaton obtained from $\mathcal{D}$ by designating q_0 as the initial state, and all states q_f such that $L(\llbracket \mathcal{A}_\Delta \rrbracket, z') \subseteq L(\mathcal{D}, q_f)$ for some $w \in (\mathsf{tr}(\tau_1) + \ldots + \mathsf{tr}(\tau_{K-1}))^k$, $k \geq k_0$, as final states. There is at least one final state because $L(\mathcal{D}) = L(\llbracket \mathcal{A}_\Delta \rrbracket)$ and $\mathcal{D}$ is deterministic. Let there be an edge in $\mathcal{D}'$ with label $\mathsf{tr}(\tau_i)$ from state q to q' whenever there is a path in $\mathcal{D}$ from q to q' over word $\mathsf{tr}(\tau_i)$. Observe that $\mathcal{D}'$ is still deterministic. Since $\rho''(w)$ is defined for any $w \in (\mathsf{tr}(\tau_1) + \ldots + \mathsf{tr}(\tau_{K-1}))^k$, $k \geq k_0$, $\mathcal{D}'$, defined over alphabet $\{\mathsf{tr}(\tau_1), \ldots, \mathsf{tr}(\tau_{K-1})\}$, is ultimately universal. But then, by Lemma 8, $\mathcal{D}'$ accepts any word in $\{\mathsf{tr}(\tau_1), \ldots, \mathsf{tr}(\tau_{K-1})\}^{K-1}$, and in particular $\mathsf{tr}(\tau_1 \ldots \tau_{K-1})$. Therefore, there is a run in $\mathcal{D}$ from $(\gamma)_{\alpha_i}$ to some state $(\gamma)_{\alpha_{i+1}}$ following $\mathsf{tr}(\tau_1 \ldots \tau_{K-1})$, which satisfies $L(\llbracket \mathcal{A}_\Delta \rrbracket, (\rho')_{\beta_{i+1}}) \subseteq L(\llbracket \mathcal{A} \rrbracket, (\gamma)_{\alpha_{i+1}})$. $\square$

Acknowledgement. I am grateful to Patricia Bouyer and Nicolas Markey for useful discussions on earlier versions of this paper.

References

1. K. Altisen and S. Tripakis. Implementation of timed automata: An issue of semantics or modeling? In FORMATS'05, LNCS 3829, p. 273–288. Springer, 2005.
2. R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.
3. R. Alur, S. La Torre, and P. Madhusudan. Perturbed timed automata. In HSCC'05, LNCS 3414, p. 70–85. Springer, 2005.
4. E. Asarin, O. Maler, A. Pnueli, and J. Sifakis. Controller synthesis for timed automata. In *In Proc. Symposium on System Structure and Control*, p. 469–474. Elsevier, 1998.
5. C. Baier, N. Bertrand, P. Bouyer, Th. Brihaye, and M. Größer. Probabilistic and topological semantics for timed automata. In FSTTCS'07, LNCS 4855, p. 179–191. Springer, 2007.
6. J. Bermond, A. Germa, M. Heydemann, and D. Sotteau. Longest paths in digraphs. *Combinatorica*, 1:337–341, 1981.
7. P. Bouyer and F. Chevalier. On conciseness of extensions of timed automata. *Journal of Automata, Languages and Combinatorics*, 10(4):393–405, 2005.
8. P. Bouyer, N. Markey, and P.-A. Reynier. Robust model-checking of linear-time properties in timed automata. In LATIN'06, LNCS 3887, p. 238–249. Springer, 2006.

9. P. Bouyer, N. Markey, and P.-A. Reynier. Robust analysis of timed automata via channel machines. In FoSSaCS'08, LNCS 4962, p. 157–171. Springer, 2008.
10. R. Brenguier and O. Sankur. Hardness of untimed language universality. *Submitted*, 2011.
11. M. De Wulf, L. Doyen, N. Markey, and J.-F. Raskin. Robust safety of timed automata. *Formal Methods in System Design*, 33(1-3):45–84, 2008.
12. M. De Wulf, L. Doyen, and J.-F. Raskin. Almost ASAP semantics: From timed models to timed implementations. *Formal Aspects of Computing*, 17(3):319–341, 2005.
13. C. Dima. Dynamical properties of timed automata revisited. In FORMATS'07, LNCS 4763, p. 130–146. Springer Berlin / Heidelberg, 2007.
14. V. Gupta, Th. A. Henzinger, and R. Jagadeesan. Robust timed automata. In HART'97, LNCS 1201, p. 331–345. Springer, 1997.
15. A. Puri. Dynamical properties of timed systems. *Discrete Event Dynamic Systems*, 10(1-2):87–113, 2000.
16. F. P. Ramsey. On a problem in formal logic. *Proc. London Math. Soc. (3)*, 30:264–286, 1930.
17. O. Sankur. Model-checking robuste des automates temporisés via les machines à canaux. Master's thesis, Ecole Normale Supérieure, 2010.