Robustness and Implementability of Timed Automata*

Martin De Wulf, Laurent Doyen**, Nicolas Markey, and Jean-François Raskin

Computer Science Departement, Université Libre de Bruxelles, Belgium

Abstract. In a former paper, we defined a new semantics for timed automata, the Almost ASAP semantics, which is parameterized by Δ to cope with the reaction delay of the controller. We showed that this semantics is implementable provided there exists a strictly positive value for the parameter Δ for which the strategy is correct. In this paper, we define the implementability problem to be the question of existence of such a Δ. We show that this question is closely related to a notion of robustness for timed automata defined in [Pur98] and prove that the implementability problem is decidable.

1 Introduction

Timed automata are an important formal model for the specification and analysis of real-time systems. Formalisms like timed automata and hybrid automata are central in the so-called *model-based development methodology for embedded controllers*. The steps underlying that methodology can be summarized as follows: (*i*) construct a (timed/hybrid automaton) model Env of the environment in which the controller will be embedded; (*ii*) make clear what is the control objective: for example, prevent the environment to enter a set of Bad states; (*iii*) design a (timed automata) model Cont of the control strategy; (*iv*) verify that $\mathsf{Reach}([\![\mathsf{Env} \parallel \mathsf{Cont}]\!]) \cap \mathsf{Bad} = \varnothing$ (where $\mathsf{Reach}([\![\mathsf{Env} \parallel \mathsf{Cont}]\!])$ denotes the set of states reachable in the transition system associated to the synchronized product of the automaton for the environment and the automaton for the controller). When Cont has been proven correct, it would be valuable to ensure that an implementation Impl of that model can be obtained in a systematic way in order to ensure the conservation of correctness, that is to ensure that $\mathsf{Reach}([\![\mathsf{Env} \parallel \mathsf{Impl}]\!]) \cap \mathsf{Bad} = \varnothing$ is obtained by construction.

Unfortunately, this is often not possible for several *fundamental* and/or *technical* reasons. First, the notion of time used in the traditional semantics of timed automata is *continuous* and defines *perfect clocks* with *infinite precision* while implementations can only access time through *digital* and *finitely precise* clocks. Second, timed automata react *instantaneously* to events and time-outs while implementations can only react within a given, usually small but not null, *reaction*

* Supported by the FRFC project "Centre Fédéré en Vérification" funded by the Belgian National Science Foundation (FNRS) under grant nr 2.4530.02

** Research fellow supported by the Belgian National Science Foundation (FNRS)

delay. Third, timed automata may describe control strategies that are *unrealistic*, like *zeno-strategies* or strategies that ask the controller *to act faster and faster* [CHR02]. For one of those three reasons, a model for a digital controller that has been proven correct may not be implementable (at all) or may not be turned systematically into an implementation that is proven correct wrt this model.

To overcome those problems, we recently proposed in [DDR04] an alternative semantics to timed automata. This semantics is called the almost ASAP semantics, AASAP for short. The AASAP-semantics of a timed automaton A, noted $[\![A]\!]^{\mathsf{AASAP}}_{\Delta}$, is a parametric semantics that leaves as a parameter $\Delta \in \mathbb{Q}^{\geq 0}$ the *reaction delay* of the controller. This semantics relaxes the classical semantics of timed automata in that it does not impose on the controller to react instantaneously but imposes on the controller to react *within Δ time units*. We have proven that a timed controller is implementable with a *sufficiently fast* hardware if and only if there exists $\Delta \in \mathbb{Q}^{>0}$ such that $\mathsf{Reach}([\![\mathsf{Env} \parallel \mathsf{Cont}]\!]_{\Delta}) \cap \mathsf{Bad} = \varnothing$. The *implementability problem* is to determine the existence of such a Δ. The decidability of that important problem is open. We will close this decidability gap in this paper.

The use of the AASAP-semantics in the verification phase can be understood intuitively as follows. When we verify a control strategy using the AASAP-semantics, we test if the proposed strategy is *robust* in the following sense: "*Is the strategy still correct if it is perturbed a little bit when executed on a device that has a finite speed and uses finitely precise clocks ?*"

In this paper, we show that this intuition relating robustness and implementability allows us to draw an interesting and important link with a paper by Puri [Pur98] and allows us to answer *positively* the open question about the decidability of the implementability problem.

Related works. In this paper, we focus on timed controllers and environments that can be modeled using timed automata. There exist related works where the interested reader will find other ideas about implementability.

In [AFILS03], Rajeev Alur et al consider the more general problem of generating code from hybrid automata, but they only sketch a solution and state interesting research questions. The work in this paper should be useful in that context.

In [AFM$^+$02,AFP$^+$03], Wang Yi et al present a tool called Times that generates executable code from timed automata models. However, they make the synchrony hypothesis and so they *assume* that the hardware on which the code is executed is infinitely fast and precise.

In [HKSP03], Tom Henzinger et al introduce a programming model for real-time embedded controllers called Giotto. Giotto is an embedded software model that can be used to specify a solution to a given control problem independently of an execution platform but which is closer to executable code than a mathematical model. So, Giotto can be seen an intermediary step between mathematical models like hybrid automata and real execution platforms.

Our paper is structured as follows. In Section 2, we recall some classical definitions related to timed automata and we introduce a general notion of enlarged semantics for those automata. In Section 3, we recall the essential notions and problems related to the AASAP-semantics, and we recall the notion of robustness as introduced by Puri in [Pur98]. In Section 4, we present a small example that illustrates the enlarged semantics and the problems that we want to solve on this semantics. In Section 5, we make formal the link between our notion of implementability and the notion of robustness introduced by Puri. In Section 6, we give a direct proof that the implementability problem is decidable. Finally, we close the paper by a conclusion.

2 Preliminaries

Definition 1 [TTS] A *timed transition system* $\mathcal{T}$ is a tuple $\langle S, \iota, \Sigma, \rightarrow\rangle$ where S is a (possibly infinite) set of states, $\iota \in S$ is the initial state, Σ is a finite set of labels, and $\rightarrow \subseteq S \times \Sigma \cup \mathbb{R}^{\geq 0} \times S$ is the transition relation where $\mathbb{R}^{\geq 0}$ is the set of positive real numbers. If $(q, \sigma, q') \in \rightarrow$ we write $q \xrightarrow{\sigma} q'$. □

A *trajectory* of a TTS $\mathcal{T} = \langle S, \iota, \Sigma, \rightarrow\rangle$ is a sequence $\pi = (s_0, t_0) \ldots (s_k, t_k)$ such that for $0 \leq i < k$, $t_i \in \mathbb{R}$, $s_i \xrightarrow{\sigma} s_{i+1}$ and either $\sigma \in \Sigma$ and $t_{i+1} = t_i$, or $\sigma \in \mathbb{R}^{>0}$ and $t_{i+1} = t_i + \sigma$. We sometimes refer to this trajectory as $\pi[t_0, t_k]$ and write $\pi(t_i)$ instead of q_i. A trajectory is *stutter-free* iff it is not the case for any i that $q_i \xrightarrow{\tau_i} q_{i+1}$ and $q_{i+1} \xrightarrow{\tau_{i+1}} q_{i+2}$ with $\tau_i, \tau_{i+1} \in \mathbb{R}^{>0}$. A state s of $\mathcal{T}$ is *reachable* if there exists a trajectory $\pi = (s_0, t_0) \ldots (s_k, t_k)$ such that $s_0 = \iota$ and $s_n = s$. The set of reachable states of $\mathcal{T}$ is noted $\mathsf{Reach}(\mathcal{T})$.

Given a set $\mathsf{Var} = \{x_1, \ldots, x_n\}$ of clocks, a *clock valuation* is a function $v \colon \mathsf{Var} \rightarrow \mathbb{R}^{\geq 0}$. In the sequel, we often say that a clock valuation is a point in $\mathbb{R}^n$. If $R \subseteq \mathsf{Var}$, then $v[R := 0]$ denotes the valuation v' such that

$$v'(x) = \begin{cases} 0 & \text{if } x \in R \\ v(x) & \text{if } x \notin R \end{cases}$$

A *closed rectangular guard* g over $\{x_1, \ldots, x_n\}$ is a set of inequalities of the form $a_i \leq x_i \leq b_i$, one for each x_i where $a_i, b_i \in \mathbb{Q}^{\geq 0} \cup \{+\infty\}$ and $a_i \leq b_i$. We write $\mathsf{Rect_c}(\mathsf{Var})$ for the set of closed rectangular guards over Var. For $\Delta \geq 0$, we define $[\![g]\!]_\Delta = \{(x_1, \ldots, x_n) \mid a_i - \Delta \leq x_i \leq b_i + \Delta\} \subseteq \mathbb{R}^n$. When $\Delta = 0$, we write $[\![g]\!]$ instead of $[\![g]\!]_0$.

We slightly modify the classical definitions related to timed automata [AD94]. In particular, guards on edges are rectangular and closed[1]. Also the value of their clock is bounded by M, the largest constant appearing in guards. The last restriction does not reduce the expressive power of timed automata.

Definition 2 [Timed automaton] A *timed automaton* is a tuple $A = \langle \mathsf{Loc}, \mathsf{Var}, q_0, \mathsf{Lab}, \mathsf{Edg}\rangle$ where

[1] In the sequel, the guards are enlarged by strictly positive parameter Δ, and so it is natural to consider them closed.

- Loc is a finite set of locations representing the discrete states of the automaton.
- $\mathsf{Var} = \{x_1, \dots, x_n\}$ is a finite set of real-valued variables.
- $q_0 = (l_0, v_0)$ where $l_0 \in \mathsf{Loc}$ is the initial location and v_0 is the initial clock valuation such that $\forall x \in \mathsf{Var} : v_0(x) \in \mathbb{N} \wedge v_0(x) \leq M$.
- Lab is a finite alphabet of labels.
- $\mathsf{Edg} \subseteq \mathsf{Loc} \times \mathsf{Loc} \times \mathsf{Rect_c}(\mathsf{Var}) \times \mathsf{Lab} \times 2^{\mathsf{Var}}$ is a set of edges. An edge (l, l', g, σ, R) represents a jump from location l to location l' with guard g, event σ and a subset $R \subseteq \mathsf{Var}$ of variables to be reset. □

We now define a family of semantics for timed automata that is parameterized by $\epsilon \in \mathbb{Q}^{\geq 0}$ (drift on clocks) and $\Delta \in \mathbb{Q}^{\geq 0}$ (imprecision on guards).

Definition 3 [Enlarged semantics of timed automata] The semantics of a timed automaton $A = \langle \mathsf{Loc}, \mathsf{Var}, q_0, \mathsf{Lab}, \mathsf{Edg} \rangle$, when the two parameters ϵ and Δ are fixed is given by the TTS $[\![A]\!]^{\epsilon}_{\Delta} = \langle S, \iota, \Sigma, \rightarrow \rangle$ where

1. $S = \{(l, v) \mid l \in \mathsf{Loc} \wedge v \colon \mathsf{Var} \rightarrow [0, M]\}$.
2. $\iota = q_0$.
3. $\Sigma = \mathsf{Lab}$.
4. The transition relation $\rightarrow$ is defined by
   (a) For the discrete transitions: $((l, v), \sigma, (l', v')) \in \rightarrow$ iff there exists an edge $(l, l', g, \sigma, R) \in \mathsf{Edg}$ such that $v \in [\![g]\!]_{\Delta}$ and $v' = v[R := 0]$.
   (b) For the continuous transitions: $((l, v), t, (l', v')) \in \rightarrow$ iff $l = l'$ and $v'(x_i) - v(x_i) \in [(1-\epsilon)t, (1+\epsilon)t]$ for $i = 1 \dots n$. □

In the sequel, we write $[\![A]\!]$ for $[\![A]\!]^0_0$, which is the classical semantics of timed automata.

We now recall some additional classical notions related to timed automata.

Let $\lfloor x \rfloor$ denote the integer part of x (the greatest integer $k \leq x$), and $\langle x \rangle$ denote its fractional part.

Definition 4 [Clock regions] A *clock region* is an equivalence class of the relation $\sim$ defined over the clock valuations in $\mathsf{Var} \rightarrow [0, M]$. We have $v \sim w$ iff all the following conditions hold:

- $\forall x \in \mathsf{Var} : \lfloor v(x) \rfloor = \lfloor w(x) \rfloor$.
- $\forall x, y \in \mathsf{Var} : \langle v(x) \rangle \leq \langle v(y) \rangle$ iff $\langle w(x) \rangle \leq \langle w(y) \rangle$.
- $\forall x \in \mathsf{Var} : \langle v(x) \rangle = 0$ iff $\langle w(x) \rangle = 0$. □

We write $]v[$ for the clock region containing v. It is easy to see that $]v[$ contains the valuations that agree with v on the integer part of the variables, and on the ordering of their fractional part and 0. The closure of the region containing v is noted $[v]$. Such a set is called a *closed region.*

Definition 5 [Region graph] Given the TTS $[\![A]\!] = \langle S, s_0, \Sigma, \rightarrow_A \rangle$ of a timed automaton A, we define the corresponding *region graph* $G = \langle \mathcal{C}, \rightarrow_G \rangle$ of A :

– $\mathcal{C} = \{(l, [v]) \mid (l, v) \in S\}$ is the set of closed regions.
– $\rightarrow_G \subseteq \mathcal{C} \times \mathcal{C}$. $((l, [v]), (l', [v'])) \in \rightarrow_G$ if $(l, v) \rightarrow_A (l', v')$ and $(l, [v]) \neq (l', [v'])$. □

This definition is meaningful since $\mathcal{C}$ is finite and whenever $(l, [v]) \rightarrow_G (l', [v'])$, for any $s \in [v]$ there exists $s' \in [v']$ such that $(l, s) \rightarrow_A (l', s')$, and for any $s' \in [v']$ there exists $s \in [v]$ such that $(l, s) \rightarrow_A (l', s')$ [AD94].

Let $W = |\mathcal{C}|$ be the total number of regions.

Definition 6 [Zones] A *zone* $Z \subseteq \mathbb{R}^n$ is a *closed set* defined by inequalities of the form

$$x_i - x_j \leq m_{ij}, \quad \alpha_i \leq x_i \leq \beta_i$$

where $1 \leq i, j \leq n$ and $m_{ij}, \alpha_i, \beta_i \in \mathbb{Z}$. □

A set of states is called a *zone-set* if it is a finite union of sets of the form $\{l\} \times Z$ where l is a location and Z is a zone.

Definition 7 [Progress cycle] A *progress cycle* in the region graph of a timed automaton is a cycle in which each clock is reset at least once. □

Assumption 8 We make the assumption that every cycle in the region graph of the timed automata we consider is a progress cycle.

This assumption, made by Puri in his paper [Pur98], is not a very restrictive assumption, since it is weaker than classical non-Zeno assumptions in the literature (for example in [AMPS98], they impose that "in every cycle in the transition graph of the automaton, there is at least one transition which resets a clock variable x_i to zero, and at least one transition which can be taken only if $x_i \geq 1$").

3 AASAP semantics and enlarged semantics

In [DDR04], we introduced the Almost ASAP semantics. This semantics relaxes the usual semantics of timed automata, its main characteristics are summarized as follows:

– any transition that can be taken becomes urgent only after a small delay Δ (which may be left as a parameter);
– a distinction is made between the occurrence of an event in the environment (sent) and in the controller (received), however the time difference between the two events is bounded by Δ;
– guards are enlarged by some small amount depending on Δ.

In the same paper, in theorem 6, we show that this semantics can be encoded using a syntactical transformation of the automaton controller and by enlarging the guards by the parameter Δ which takes its value in the positive rationals. So we can study the AASAP-semantics of $\mathsf{Env} \parallel \mathsf{Cont}$ by considering the semantics $[\![\mathsf{Env} \parallel \mathsf{Cont}']\!]^0_\Delta$ where Cont' is obtained syntactically from Cont. So in the rest of this paper, we will consider the Δ-enlarged semantics instead of the AASAP-semantics.

In this previous work, we have shown that the AASAP-semantics and so the Δ-enlarged semantics allow us to reason about the implementability of a control strategy defined by a timed automaton. The problems that we want to solve algorithmically on the Δ-enlarged semantics are the following ones:

- [Fixed] given a zone-set of Bad states, Env and Cont, decide whether $\mathsf{Reach}([\![\mathsf{Env} \parallel \mathsf{Cont}']\!]^0_\Delta) \cap \mathsf{Bad} = \varnothing$ for a given fixed value of $\Delta \in \mathbb{Q}^{>0}$;
- [Existence] given a zone-set of Bad states, Env and Cont, decide whether there exists $\Delta \in \mathbb{Q}^{>0}$ such that $\mathsf{Reach}([\![\mathsf{Env} \parallel \mathsf{Cont}']\!]^0_\Delta) \cap \mathsf{Bad} = \varnothing$. This is also called the implementability problem.
- [Maximization] given a zone-set of Bad states, Env and Cont, compute the least upper bound of the set of $\Delta \in \mathbb{Q}^{>0}$ such that $\mathsf{Reach}([\![\mathsf{Env} \parallel \mathsf{Cont}']\!]^0_\Delta) \cap \mathsf{Bad} = \varnothing$. Intuitively, this gives us the information about the slowest hardware that can implement correctly the strategy.

To solve the fixed version, we use the usual reachability algorithm for timed automaton defined in [AD94]. To solve the maximization version (in an approximative way), we observe that for any timed automaton A, any two positive rational numbers Δ_1, Δ_2, if $\Delta_1 \leq \Delta_2$ then $\mathsf{Reach}([\![A]\!]^0_{\Delta_1}) \subseteq \mathsf{Reach}([\![A]\!]^0_{\Delta_2})$. So, given a tolerance $\eta \in \mathbb{Q}^{>0}$, the maximal value of Δ can be approached by η as follows: assuming Bad is reachable in $[\![A]\!]^0_{\Delta=1}$, it suffices to solve the [Fixed] problems with values $\Delta_i = i\eta$ $(0 \leq i \leq \lceil \frac{1}{\eta} \rceil)$, and take as approximation of the optimal the value Δ_i such that the answer of the [Fixed] problem is Yes for Δ_i and No for Δ_{i+1}, which can be found more efficiently with a dichotomy search. The decidability of the implementability problem is established in the next sections.

To achieve this, we draw a strong link with a robust semantics defined by Puri in [Pur98]. In that paper, Puri shows that the traditional reachability analysis defined in [AD94] is not correct when the clocks drift even by a very small amount. He then reformulates the reachability problem as follows: given a timed automaton A, instead of computing $\mathsf{Reach}([\![A]\!]^0_0)$, he proposes an algorithm that computes $\cap_{\epsilon \in \mathbb{Q}^{>0}} \mathsf{Reach}([\![A]\!]^\epsilon_0)$. When A is clear from the context, this set is denoted by R^*_ϵ. This is the set of states that can be reached when the clocks drift by an infinitesimally small amount. He shows that this set has nice robustness properties with respect to modeling errors. In particular, he establishes that if the clocks are drifting, then guards can be checked with some small imprecisions (see [Pur98] for details).

In our paper, in order to make the link with the implementability problem, we study a variant of this robust semantics where only small imprecisions on guards checking are allowed: the set of reachable states in this semantics is the set $\cap_{\Delta \in \mathbb{Q}^{>0}} \mathsf{Reach}([\![A]\!]^0_\Delta)$. When A is clear from the context, this set is abbreviated by R^*_Δ. We first show that for any timed automaton A, any zone-set Bad, we have that: $\cap_{\Delta\in\mathbb{Q}^{>0}}\mathsf{Reach}([\![A]\!]^0_\Delta) \cap \mathsf{Bad} = \varnothing$ iff there exists $\Delta \in \mathbb{Q}^{>0}$ such that $\mathsf{Reach}([\![A]\!]^0_\Delta) \cap \mathsf{Bad} = \varnothing$. After, we establish that the algorithm proposed by Puri to compute the set $\cap_{\epsilon\in\mathbb{Q}^{>0}}\mathsf{Reach}([\![A]\!]^\epsilon_0)$ is also valid to compute the set of states $\cap_{\Delta\in\mathbb{Q}^{>0}}\mathsf{Reach}([\![A]\!]^0_\Delta)$. As corollaries, we obtain that $\cap_{\epsilon\in\mathbb{Q}^{>0}}\mathsf{Reach}([\![A]\!]^\epsilon_0) = \cap_{\Delta\in\mathbb{Q}^{>0}}\mathsf{Reach}([\![A]\!]^0_\Delta)$ and so we obtain an algorithm to decide the implementability problem.

The proofs of our results follow the general ideas of the proofs of Puri and are based on the structure of limit cycles of timed automata (a fundamental notion introduced by Puri) but we needed new techniques to treat the imprecisions on guards instead of the drifts of clocks as in the original paper. Also the proofs in the paper of Puri are not always convincing, so we reproved a large number of his lemmas that are needed to establish our proof and had to correct one of them. The interested reader is invited to consult the appendix of this paper for all the proofs.

4 Example

Consider automaton A of Fig. 1. We examine two cases : $\alpha = 2$ and $\alpha = 3$. For both cases, the reachable states of the classical semantics $[\![A]\!]$ are the same and are depicted in Fig. 2 (the points $v_0 \dots v_7$ will be used later in the paper). The safety property we want to verify is that the location *err* is not reachable. Note that in the classical semantics this is true in both cases.

Consider now the enlarged semantics $[\![A]\!]^0_\Delta$ with $\Delta > 0$. In this semantics, guards are enlarged by the amount Δ. The edge from l_1 to l_2 has the guard $a \leq 2+\Delta$ and the edge from l_2 to l_1 has the guard $b \geq 2-\Delta$. Starting from the initial state $(l_1, a = 1, b = 0)$, the jump to l_2 can occur Δ time units later, so that the states $(l_2, a = 0, b \leq 1+\Delta)$ are reached. Similarly, the transition from l_2 back to l_1 is enabled Δ time units earlier and the states $(l_1, a \geq 1-2\Delta, b = 0)$ can be reached. By iterating the cycle, the states $(l_1, a \geq 1 - 2k\Delta, b = 0)$ and $(l_2, a = 0, b \leq 1 + (2k-1)\Delta)$ are reachable. So, for any $\Delta > 0$ some new states are reachable in $[\![A]\!]^0_\Delta$ that were not reachable in the classical semantics. Those states are represented in Fig. 3.

From those new states that become reachable in location l_2, if $\alpha = 3$, the location *err* remains unreachable but if $\alpha = 2$ it becomes reachable.

Clearly, from this example, one sees that a correct timed system (in the sense of the classical semantics) could have a completely different (and potentially bad) behavior due to an infinitesimally small inaccuracy in testing of the clocks (which is unavoidable since the clocks are discrete in embedded systems). This is the case for the automaton of figure 1. When $\alpha = 2$, the classical semantics is not robust since even the slightest error in guard checking allows new discrete transition to be taken. In other words, there is no strictly positive value for the parameter Δ that still ensures the safety property for $[\![A]\!]^0_\Delta$. Systems with such features cannot have a correct implementation because their correctness relies on the idealization of the mathematical model.

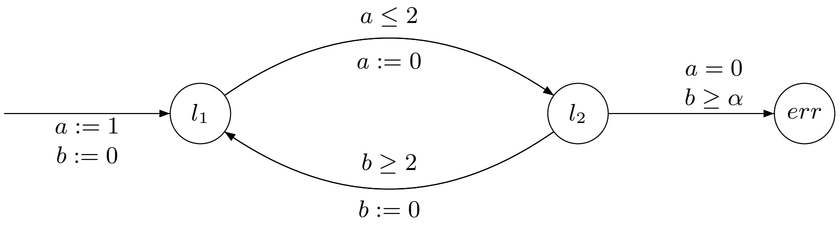


Fig. 1. A timed automaton A.

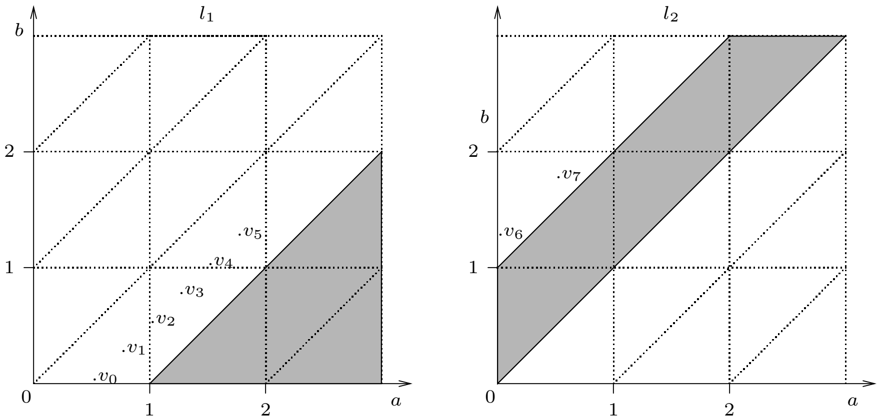


Fig. 2. $\mathsf{Reach}([\![A]\!])$ for the timed automaton A of Fig. 1.

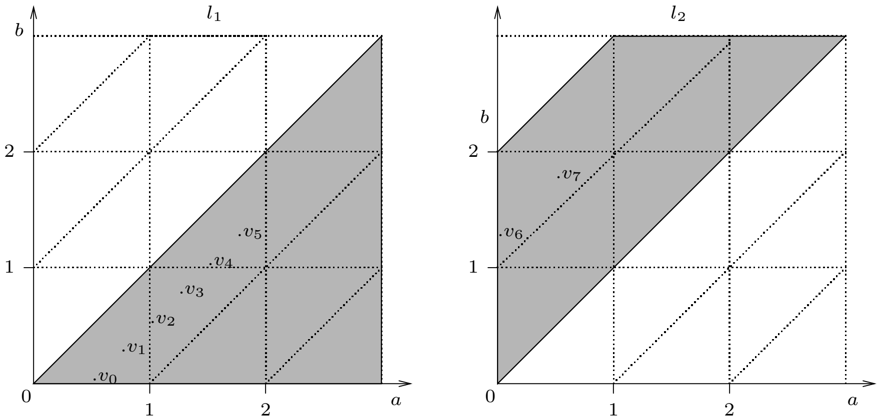


Fig. 3. The set R^*_Δ for the timed automaton A of Fig. 1.

But this is not always the case: for the same automaton when $\alpha = 3$, no more discrete transitions are possible in the enlarged semantics than in the classical one. In this case, we can answer positively to the question "Is there a strictly positive value for parameter Δ that allows the enlarged semantics to still satisfy the safety property?". And indeed, we can prove that any value $\Delta < \frac{1}{3}$ is satisfying.

5 Linking robustness and implementability

The classical semantics of timed automaton A is $[\![A]\!]_{\Delta=0}^{\epsilon=0}$, which is a mathematical idealization of how we expect an implementation would behave: it makes the hypothesis that the hardware is perfectly precise and infinitely fast. Unfortunately, the execution of a timed automaton on a real hardware cannot be considered as ideal in the mathematical sense. It is thus an interesting question to know whether a small drift or imprecision of the clocks could invalidate some properties satisfied by the classical semantics. Drifts in clocks have been studied in [Pur98]. We are interested in studying imprecisions in the evaluation of the guards since it is directly connected to the question of implementability, as explained above. The main result of this paper is the following.

Theorem 9 *There exists an algorithm that decide for any timed automaton A, any zone-set* Bad *if there exists $\Delta \in \mathbb{Q}^{>0}$ such that* $\mathsf{Reach}([\![A]\!]_{\Delta}^{0}) \cap \mathsf{Bad} = \varnothing$.

To prove Theorem 9, we show that the set Bad intersects R_{Δ}^{*} iff it intersects $\mathsf{Reach}([\![A]\!]_{\Delta}^{0})$ for some $\Delta > 0$. As shown in the next section, Algorithm 1 computes R_{Δ}^{*}. Consequently, the implementability problem is decidable.

Theorem 10 *For any timed automaton A, any zone-set* Bad, $R_{\Delta}^{*} \cap \mathsf{Bad} = \varnothing$ *iff* $\exists \Delta > 0 : \mathsf{Reach}([\![A]\!]_{\Delta}^{0}) \cap \mathsf{Bad} = \varnothing$.

The proof of Theorem 10 relies on two intermediate lemmas, one of which corrects a wrong claim in [Pur98]: it gives a bound on the distance between two zones with empty intersection. This bound is claimed to be $\frac{1}{2}$. We show that $\frac{1}{n}$ where n is the dimension of the space is the tightest bound. However the final results of [Pur98] are not deeply affected by this mistake. The distance considered is defined by $d_{\infty}(x, y) = \|x - y\|_{\infty} = \max_{1 \leq i \leq n}(|x_i - y_i|)$. Let us reformulate that lemma.

Lemma 11 (Corrected from [Pur98, Lemma 6.4]) *Let $Z_1 \subseteq \mathbb{R}^n$ and $Z_2 \subseteq \mathbb{R}^n$ be two zones such that $Z_1 \cap Z_2 = \varnothing$. Then, for any $x \in Z_1$ and $y \in Z_2$, $d_{\infty}(x, y) \geq \frac{1}{n}$. This bound is tight.*

In the sequel, when a distance d or a norm $\| \ \|$ is used, we always refer to d_{∞} and $\| \ \|_{\infty}$. The following lemma relies on the theory of real numbers and the basics of topology.

Algorithm 1: Algorithm from [Pur98] for computing $R^*_\epsilon(A)$ for a timed automaton A.

Data : A timed automaton $A = \langle \mathsf{Loc}, \mathsf{Var}, q_0, \mathsf{Lab}, \mathsf{Edg} \rangle$
Result : The set $J^* = R^*_\epsilon$
begin
  1. Construct the region graph $G = (R_A, \longrightarrow_A)$ of A ;
  2. Compute $SCC(G)$ = strongly connected components of G;
  3. $J^* \leftarrow [(q_0)]$;
  4. $J^* \leftarrow \mathsf{Reach}(G, J^*)$;
  5. **if** for some $S \in SCC(G)$, $S \not\subseteq J^*$ and $J^* \cap S \neq \varnothing$ **then**
    $J^* := J^* \cup S$;
    Goto 4 ;
end

Lemma 12 *Let $A_\Delta (\Delta \in \mathbb{R}^{>0})$ be a collection of sets such that $A_{\Delta_1} \subseteq A_{\Delta_2}$ if $\Delta_1 \leq \Delta_2$. Let $A = \bigcap_{\Delta > 0} A_\Delta$ be nonempty. If $d(A, B) > 0$, then there exists $\Delta > 0$ such that $A_\Delta \cap B = \varnothing$.*

Both lemmas are proved in Appendix A.

Proof of theorem 10. If $R^*_\Delta \cap \mathsf{Bad} = \varnothing$, since R^*_Δ and Bad are unions of sets of the form $\{l\} \times Z_l$ where Z_l is a zone, Lemma 11 applies and we have $d(R^*_\Delta, \mathsf{Bad}) > 0$. From Lemma 12, we obtain that there exists $\Delta > 0$ such that $\mathsf{Reach}([\![A]\!]^0_\Delta) \cap \mathrm{Bad} = \varnothing$.

If there exists $\Delta > 0$ such that $\mathsf{Reach}([\![A]\!]^0_\Delta) \cap \mathrm{Bad} = \varnothing$, then trivially $R^*_\Delta \cap \mathsf{Bad} = \varnothing$. ■

6 Algorithm for computing R^*_Δ

In this section, we prove that the algorithm proposed in [Pur98] computes R^*_Δ. This implies that $R^*_\epsilon = R^*_\Delta$. The algorithm is shown as Algorithm 1.

Let us first examine how Algorithm 1 performs on the example of section 4. In the region graph of the timed automaton of Fig. 1, there is a cycle that runs from valuation v_0 to itself through v_1 to v_7 (see Fig. 2). Thus there is a cycle through the regions containing valuations v_0 to v_7. Furthermore, these regions have an intersection with the set of reachable states in the classical semantics (in gray). Indeed since we consider closed regions, the intersection of two adjacent regions is not empty. Since those regions form a strongly connected component of the region graph and their intersection with the reachable states in the classical semantics is not empty, the algorithm adds all those regions to the set J^*.

One can check that all regions of R^*_Δ for the automaton A of figure 1 will be correctly added by Algorithm 1 (see figure 3).

In the rest of this section, we prove that this algorithm computes R^*_Δ, by proving $J^* \subseteq R^*_\Delta$ on the one hand, and $R^*_\Delta \subseteq J^*$ on the other hand.

6.1 Limit cycles

This section studies the behavior of limit cycles. A *limit cycle* of a timed automaton A is a trajectory $\pi = (q_0, t_0)(q_1, t_1) \dots (q_k, t_k)$ of $[\![A]\!]$ such that $t_k > t_0$ and $q_k = q_0$. As suggested in [Pur98], given a progress cycle in the region graph and a region on this cycle, we focus on the subset of points of this region having a limit cycle. We first define this subset:

Definition 13 [See [Pur98, Section 7.1]] Consider a cyclic path $p = p_0\, p_1\, \dots\, p_N$ with $p_N = p_0$ in the region graph of a timed automaton A. We define the *return map* $R_p \colon 2^{p_0} \to 2^{p_0}$ by $R_p(S) = \cup_{q \in S} R_p(\{q\})$ for $S \subseteq p_0$, and, for singletons,

$$R_p(\{q_0\}) = \left\{ q_N \;\middle|\; \begin{array}{l} \text{there exists a trajectory } \pi \text{ in } [\![A]\!] \text{ s.t.} \\ \pi = (q_0, t_0)(q_1, t_1) \dots (q_N, t_N) \text{ and } \forall i.\ q_i \in p_i \end{array} \right\}.$$

The set $L_{i,p}$ of points which can return back to themselves after i cycles through p, is defined as follows: $L_{i,p} = \{q \mid q \in R_p^i(q)\}$. The set of points with limit cycles through p is $L_p = \cup_{i \in \mathbb{N}} L_{i,p}$. □

In the sequel, we write R or L instead of R_p or L_p when the path p is clear from the context. The interesting property of L_p is that it is always reachable from any valuation in p:

Theorem 14 ([Pur98, Lemma 7.10**])** *Let $p = p_0\, p_1\, \dots\, p_N$ be a cycle in G. Then for any $z \in p_0$, there exists $z', z'' \in L$ s.t. there exist trajectories in $[\![A]\!]$ from z to z' and from z'' to z.*

The proof proposed by Puri is quite sketchy, and we propose a complete proof in Appendix B.

6.2 Soundness of Algorithm 1: $J^* \subseteq R^*_\Delta$

Let $r \in \mathbb{R}^{\geq 0}$ and $x \in \mathbb{R}^n$. The *closed ball* of radius r centered in x is the set $B(x, r) = \{x' \mid d(x, x') \leq r\}$. The following Lemma shows how a trajectory may be modified when enlarging guards in timed automata:

Lemma 15 *Let A be a timed automaton with n clocks, $\Delta \in \mathbb{Q}^{>0}$, and $\delta = \frac{\Delta}{n}$. Let $p = p_0\, p_1\, p_2 \dots p_N$ be a progress cycle in the region graph. Let u be a valuation in p_0 having a limit cycle, i.e. for which there exists a trajectory $\pi[0, T]$ in $[\![A]\!]_0^0$ following p and with $\pi(0) = \pi(T) = u$. Let $v \in p_0 \cap B(u, \delta)$ be a neighbor valuation. Then there exists a trajectory from u to v in $[\![A]\!]_\Delta^0$.*

The technical proof is in Appendix C. Intuitively, the result is proved by slightly anticipating or delaying the transition dates when a clock is reset.

Consider for instance the following path:

$$(x = a, y = b) \xrightarrow{\tau_1} \xrightarrow[x:=0]{} (x = 0, y = b + \tau_1) \xrightarrow{\tau_2} (x = a', y = b').$$

with $a' = \tau_2$ and $b' = b + \tau_1 + \tau_2$. By slightly modifying the continuous transition labels τ_i, we get

$$(x = a, y = b) \xrightarrow[x:=0]{\tau_1 - \delta} (x = 0, y = b + \tau_1 - \delta) \xrightarrow{\tau_2 + \delta} (x = a' + \delta, y = b').$$

Thus the final value of clock y is identical in both cases, while the final value of x has been slightly modified. By carefully repeating this procedure, we can independently modify the final valuations of each clock. This corresponds to the idea behind Lemma 15.

Theorem 16 *Let $p = p_0\, p_1\, \dots\, p_N$ be a progress cyclic path in the region graph of A, and let x and y be two valuations in p_0. For any $\Delta \in \mathbb{Q}^{>0}$, there exists a trajectory from x to y in $[\![A]\!]^0_\Delta$.*

Proof. From Lemma 14, there exists $u, v \in L$ s.t. $x \to u$ and $v \to y$ in $[\![A]\!]$. Then u satisfies the conditions of Lemma 15, and thus any $w \in L$ "close to" u is reachable from u in $[\![A]\!]^0_\Delta$. Since L is convex, by recursively applying the previous argument, any $w \in L$ is reachable from u in $[\![A]\!]^0_\Delta$. In particular, v is reachable from u in $[\![A]\!]^0_\Delta$, and y is reachable from x in $[\![A]\!]^0_\Delta$. ■

As a consequence:

Theorem 17 *The set J^* computed by Algorithm 1 is a subset of R^*_Δ, i.e. $J^* \subseteq R^*_\Delta$.*

Proof. Let $\Delta > 0$. If a set of regions J is a subset of $\mathsf{Reach}([\![A]\!]^0_\Delta)$, then so is the set of reachable regions from J in the region graph G. Moreover, whenever a state of J appears in a cyclic region path p, then Theorem 16 ensures that any state in that region path is reachable in $[\![A]\!]^0_\Delta$. Thus $J \cup p \subseteq \mathsf{Reach}([\![A]\!]^0_\Delta)$. Since J^* is built by successively applying the above two operations, this ensures that $J^* \subseteq \mathsf{Reach}([\![A]\!]^0_\Delta)$. This holds for any $\Delta > 0$, thus $J^* \subseteq R^*_\Delta$. ■

6.3 Completeness of Algorithm 1: $R^*_\Delta \subseteq J^*$

To prove the completeness of Algorithm 1, we need to better understand the relationship between trajectories of $[\![A]\!]^0_\Delta$ and those of $[\![A]\!]$. In particular, Theorem 18 states that any trajectory π' of $[\![A]\!]^0_\Delta$ can be approached by a trajectory π of $[\![A]\!]$, making the same discrete transitions and passing by the same regions.

Theorem 18 *For any distance $\delta \in \mathbb{R}^{>0}$, for any number of steps $k \in \mathbb{N}$, there exists $\Delta \in \mathbb{Q}^{>0}$ such that for any k-step trajectory $\pi' = (q'_0, t'_0) \dots (q'_k, t'_k)$ of $[\![A]\!]^0_\Delta$, there exists a k-step trajectory $\pi = (q_0, t_0) \dots (q_k, t_k)$ of $[\![A]\!]$ such that for any position i $(0 \leq i \leq k)$*

- *$q_i \in [q'_i]$, that is the two trajectories cross the same regions,*

– *and the distance between corresponding states q_i and q'_i is bounded by δ, that is $q_i = (l_i, v_i)$, $q'_i = (l'_i, v'_i)$ where $l_i = l'_i$ and $\|v_i - v'_i\| \leq \delta$.*

The complete proof of this theorem is given in Appendix D. Below, we give the general ideas underlying the proof. It uses the following lemma, stating that if v is reachable from u by letting time elapse, then any point $x \in [u]$ in the neighborhood of u can reach some point $y \in [v]$ in the neighborhood of v. More precisely:

Lemma 19 *Suppose $r \rightarrow_G r'$ in the region graph G of a timed automaton, and $u \xrightarrow{\tau} v$ with $u \in r$, $v \in r'$ and $\tau \in \mathbb{R}^{\geq 0}$. Then for any $x \in B(u, R) \cap r$, there exists $y \in B(v, 2R) \cap r'$ such that $x \xrightarrow{\tau'} y$ for some τ' with $|\tau - \tau'| \leq R$.*

The proof of Lemma 19 is given in Appendix D.

Proof (Sketch) of theorem 18.

The outline of the proof is as follows. We define a "simulation" relation $\mathcal{R}$ between valuations, which is a subset of $\mathbb{R}^n \times \mathbb{N} \times \mathbb{R}^n$ and we write $\mathcal{R}_a(x', x)$ if $(x', a, x) \in \mathcal{R}$. We show that

$$\text{If } \begin{cases} a \leq k \\ \mathcal{R}_a(x', x) \\ x' \xrightarrow{t'} y' \xrightarrow{\sigma} z' \text{ in } [\![A]\!]^0_\Delta \end{cases} \quad \text{then } \exists y, z : \begin{cases} x \xrightarrow{t} y \xrightarrow{\sigma} z \text{ in } [\![A]\!] \\ \mathcal{R}_{a+1}(z', z) \\ \|z - z'\| \leq \|y - y'\| \leq \delta \end{cases}$$

The relation $\mathcal{R}_a$ takes into account the current position in the trajectory: if x' is the a^{th} state in the trajectory, then x is in a neighborhood of size parameterized by a. This size increases with a, because the longer is the trajectory, the greater is the possible deviation (Fig. 4). This is not a classical simulation relation because it only works for a fixed trajectory length k (which is sufficient for proving Theorem 18).

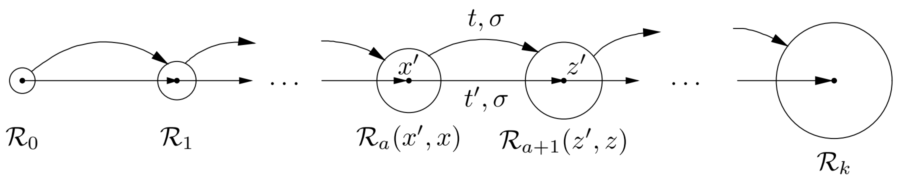


Fig. 4. Simulation of the trajectory π' of $[\![A]\!]^0_\Delta$.

Intuitively, $\mathcal{R}$ links a valuation x with a valuation x' if whenever two clock have *close* fractional parts in x' (for a special distance $\mathcal{D}$), those clocks have *identical* fractional parts in x.

Fig. 5 illustrates the necessity of this choice: consider the trajectory $(x', 0)$ $(y', t')(z', t')$ of $[\![A]\!]^0_\Delta$. We must construct a trajectory $(x, 0)(y, t)(z, t)$ of $[\![A]\!]$ taking the same discrete transition (labeled with σ). In the enlarged semantics, the guard on this transition is satisfied by y' (*i.e.* $y' \in [\![g]\!]_\Delta$).

The only possible choice for y is to both lie in the region $[y']$ and in $[\![g]\!]_0$ as shown in the figure. This imposes the choice of x on the diagonal (*i.e.* with identical fractional parts for x_i and x_j).

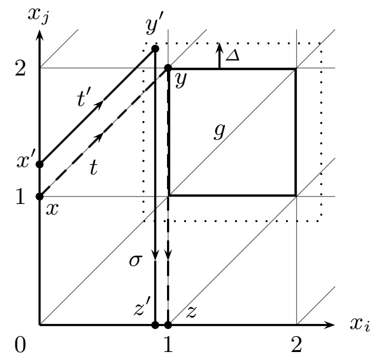


Fig. 5. Simulating $x' \xrightarrow{t'} y' \xrightarrow{\sigma} z'$ enforces to take x with $\langle x_i \rangle = \langle x_j \rangle$. This is because $\langle x'_i \rangle$ and $\langle x'_j \rangle$ are too close.

$\mathcal{R}$ is formally defined in Appendix D, and it is shown that the Δ can be chosen such that $\Delta < \frac{\delta}{2^k(2+4n)}$. ∎

The following theorem is a direct consequence of Theorem 18. It says that points in J^* cannot reach points that are more than distance α away from J^* in $[\![A]\!]^0_\Delta$ for sufficiently small Δ.

Theorem 20 *Suppose $x \in J^*$ and there exists a trajectory from x to y in $[\![A]\!]^0_\Delta$ with $\Delta < \frac{\alpha}{2^{W+1}(2+4n)}$ where $\alpha < \frac{1}{2n}$, and W is the total number of regions. Then $d(y, J^*) < \alpha$.*

The proof is given in Appendix D.

Finally, we can establish the completeness of Algorithm 1.

Theorem 21 *The set J^* computed by Algorithm 1 contains R^*_Δ, i.e. $R^*_\Delta \subseteq J^*$.*

Proof. Since J^* is a closed set containing the initial state, from Theorem 20, for any $y \in \mathsf{Reach}([\![A]\!]^0_\Delta$, $d(y, J^*) \leq \alpha$ where α can be made arbitrarily small. It follows that $R^*_\Delta \subseteq J^*$. ∎

7 Conclusion

In this paper, we have shown that a notion of robustness defined by Puri [Pur98] is closely related to a notion of implementability that we recently introduced in [DDR04]. Making this link formal allowed us to show that our notion of implementability is decidable for the class of timed automata. To establish this link, we have proved that the algorithm proposed by Puri computes the set of reachable states of timed automata where guards are enlarged by an infinitesimally small rational value. The existence of such a value implies the implementability as shown in our previous paper. The proofs of the decidability result relies on non trivial adaptations of the main ideas underlying the study of drift in the rate of clocks made by Puri.

The algorithm that is used to check implementability manipulates strongly connected components of the region graph. It can be seen as defining exact accelerations of cycles of the timed automaton.

We will work in the future on making those accelerations practical and as a consequence, we will work on to turn the theoretical algorithm proposed in this paper into a practical one. If we succeed in this task, the results of this paper and of our previous paper [DDR04] will allow us to propose a practical procedure to produce provably correct code from proved correct controller modeled as timed automata.

References

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[AFILS03] R. Alur, J. Kim F. Ivancic, I. Lee, and O. Sokolsky. Generating embedded software from hierarchical hybrid models. In *ACM Symposium on Languages, Compilers, and Tools for Embedded Systems*, 2003.

[AFM+02] Tobias Amnell, Elena Fersman, Leonid Mokrushin, Paul Pettersson, and Wang Yi. Times: A tool for modelling and implementation of embedded systems. In J.-P. Katoen and P. Stevens, editors, *Proc. Of The 8 Th International Conference On Tools And Algorithms For The Construction And Analysis Of Systems*, number 2280 in Lecture Notes In Computer Science, pages 460–464. Springer–Verlag, 2002.

[AFP+03] Tobias Amnell, Elena Fersman, Paul Pettersson, Hongyan Sun, and Wang Yi. Code synthesis for timed automata. *Nordic Journal of Computing(NJC)*, 9, 2003.

[AMPS98] E. Asarin, O. Maler, A. Pnueli, and J. Sifakis. Controller synthesis for timed automata. In *Proc. System Structure and Control*. Elsevier, 1998.

[CHR02] F. Cassez, T.A. Henzinger, and J.-F. Raskin. A comparison of control problems for timed and hybrid systems. In *HSCC 02: Hybrid Systems—Computation and Control*, Lecture Notes in Computer Science 2289, pages 134–148. Springer-Verlag, 2002.

[DDR04] M. De Wulf, L. Doyen, and J.-F. Raskin. Almost ASAP semantics: From timed models to timed implementations. In *HSCC 04: Hybrid Systems—Computation and Control*, Lecture Notes in Computer Science 2993, pages 296–310. Springer-Verlag, 2004.

[Dil90] David Dill. Timing assumptions and verification of finite-state concurrent systems. In *Proc. Int. Workshop Automatic Verification Methods for Finite State Systems (CAV'89), Grenoble, June 1989*, volume 407 of *Lecture Notes in Computer Science*, pages 197–212. Springer-Verlag, 1990.

[HKSP03] T.A. Henzinger, C.M. Kirsch, M.A. Sanvido, and W. Pree. From control models to real-time code using GIOTTO. *IEEE Control Systems Magazine*, 23(1):50–64, 2003.

[Pur98] Anuj Puri. Dynamical properties of timed automata. In *Proceedings of Formal Techniques in Real-Time and Fault-Tolerant Systems, 5th International Symposium, FTRTFT'98, Lyngby, Denmark, September 14-18, 1998*, volume 1486 of *Lecture Notes in Computer Science*, pages 210–227. Springer, 1998.

A Proofs of Lemmas 11 and 12

Lemma 11 (Corrected from [Pur98, Lemma 6.4]**)** *Let $Z_1 \subseteq \mathbb{R}^n$ and $Z_2 \subseteq \mathbb{R}^n$ be two zones such that $Z_1 \cap Z_2 = \varnothing$. Then, for any $x \in Z_1$ and $y \in Z_2$, $d_\infty(x, y) \geq \frac{1}{n}$. This bound is tight.*

Let us recall the definition of the 1-norm and the ∞-norm :

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i| \qquad\qquad \|x\|_1 = \sum_{i=1}^{n} |x_i|$$

Proof. First, we show that $\frac{1}{n}$ is a lower bound. Clearly, for any $v \in \mathbb{R}^n$, $\|v\|_1 \leq n \cdot \|v\|_\infty$. We prove that $\|x - y\|_1 \geq 1$, which entails our result.

Consider two zones given under the form of Difference Bound Matrices [Dil90]: $Z_1 \equiv (m_{i,j})$ and $Z_2 \equiv \left(m'_{i,j}\right)$. Since $Z_1 \cap Z_2 = \varnothing$, there must exist a "negative cycle"

$$m^{(\prime)}_{i_1,i_2} + m^{(\prime)}_{i_2,i_3} + m^{(\prime)}_{i_3,i_4} + \cdots + m^{(\prime)}_{i_p,i_1} \leq -1.$$

We may assume that at least one $m_{i,j}$ and one $m'_{i',j'}$ appears in this sum, since otherwise Z_1 (or Z_2) is empty and the result holds vacuously.

Since $m_{a,b} + m_{b,c} \geq m_{a,c}$ for any a, b, c, we can merge any two *consecutive* $m_{i,j}$'s into one while keeping the inequality. The same holds for $m'_{i',j'}$, and we can thus asume that $m_{i,j}$ and $m'_{i',j'}$ alternate in the sum above (starting with m_{i_1,i_2}, say).

Pick $x \in Z_1$ and $y \in Z_2$. Then

$$(x_{i_2} - x_{i_1}) + (y_{i_3} - y_{i_2}) + (x_{i_4} - x_{i_3}) + \cdots + (y_{i_1} - y_{i_p}) \leq -1.$$

Terms can be rearranged in this sum, giving

$$(y_{i_1} - x_{i_1}) - (y_{i_2} - x_{i_2}) + (y_{i_3} - x_{i_3}) - \cdots - (y_{i_p} - x_{i_p}) \leq -1.$$

If $i_k = 0$ for some k, then $x_{i_k} - y_{i_k} = 0$. Thus, we assume $1 \leq i_k \leq n$. We take the absolute value, and apply the triangle inequality:

$$\begin{aligned} 1 &\leq \left|(y_{i_1} - x_{i_1}) - (y_{i_2} - x_{i_2}) + (y_{i_3} - x_{i_3}) - \cdots - (y_{i_p} - x_{i_p})\right| \\ &\leq |(y_{i_1} - x_{i_1})| + |(y_{i_2} - x_{i_2})| + |(y_{i_3} - x_{i_3})| + \cdots + |(y_{i_p} - x_{i_p})| \\ &\leq \|x - y\|_1. \end{aligned}$$

Now, let us show that this bound is tight. Consider the zones $Z_1, Z_2 \subseteq \mathbb{R}^n$ defined by the following equations:

– If n is odd

$$Z_1 \equiv \begin{cases} x_1 = 1 \\ x_{2i} - x_{2i+1} = 0 \;\; 1 \leq i \leq \frac{n-1}{2} \end{cases}$$

$$Z_2 \equiv \begin{cases} x_n = 0 \\ x_{2i-1} - x_{2i} = 0 \;\; 1 \leq i \leq \frac{n-1}{2} \end{cases}$$

– If n is even

$$Z_1 \equiv \begin{cases} x_1 = 1 \\ x_{2i} - x_{2i+1} = 0 \ 1 \leq i \leq \frac{n}{2} - 1 \\ x_n = 0 \end{cases}$$

$$Z_2 \equiv \left\{ x_{2i-1} - x_{2i} = 0 \ 1 \leq i \leq \frac{n}{2} \right.$$

We have $Z_1 \cap Z_2 = \varnothing$: combining equations of Z_1 and Z_2 yields $\forall i, j : x_i = x_j$, which leads to a contradiction since $x_1 = 1$ and $x_n = 0$. On the other hand, let $P(1, \frac{n-2}{n}, \frac{n-2}{n}, \frac{n-4}{n}, \frac{n-4}{n}, \dots)$ and $Q(\frac{n-1}{n}, \frac{n-1}{n}, \frac{n-3}{n}, \frac{n-3}{n}, \frac{n-5}{n}, \dots)$ (take the first n coordinates). It is easy to check that $P \in Z_1$ and $Q \in Z_2$, while $d_\infty(P, Q) = \mathsf{max}(\frac{1}{n}, \dots, \frac{1}{n}) = \frac{1}{n}$. ∎

Before proving Lemma 12, we claim the following two results:

Lemma 22 *If $d(A, B) > 0$, then $A \cap B = \varnothing$.*

Lemma 23 *If $A \subseteq B$, then $d(A, C) \geq d(B, C)$ for any C.*

We can now prove Lemma 12:

Lemma 12 *Let $A_\Delta (\Delta \in \mathbb{R}^{>0})$ be a collection of sets such that $A_{\Delta_1} \subseteq A_{\Delta_2}$ if $\Delta_1 \leq \Delta_2$. Let $A = \bigcap_{\Delta > 0} A_\Delta$ be nonempty. If $d(A, B) > 0$, then there exists $\Delta > 0$ such that $A_\Delta \cap B = \varnothing$.*

Proof. Let $\Delta_i = \frac{1}{i}$ $(i \geq 1)$. Define the sequence $\left(d_i\right)_{i \geq 1}$ as follows: $d_i = d(A_{\Delta_i}, B)$. We use Lemma 23 to show that the sequence $\left(d_i\right)$ is increasing and bounded (we can exclude the case $d(A, B) = \infty$ since it makes the lemma trivially true).

Since $A_{\Delta_{i+1}} \subseteq A_{\Delta_i}$, we have $d_i \leq d_{i+1}$.
Since $A \subseteq A_{\Delta_i}$, we have $0 \leq d_i \leq d(A, B)$.

Then the sequence $\left(d_i\right)$ converges (to $d := \sup\{d_i\} \leq d(A, B)$). Assume $d \neq d(A, B)$ (then $d < d(A, B)$). For $\epsilon < d(A, B) - d$, there exists N such that $\forall i \geq N : d_i < d + \epsilon < d(A, B)$. Then, for any $i \geq N$, there exists $x \in A_{\Delta_i}$ such that $d(\{x\}, B) < d(A, B)$, and consequently $x \notin A$. Since for any $0 < \Delta \leq \frac{1}{N}$, there exists $i \geq N$ such that $A_{\frac{1}{i}} \subseteq A_\Delta$, we have for any $\Delta \leq \frac{1}{N}$ that there exists $x \in A_\Delta$ such that $x \notin A$. Since A is also equal to $\bigcap_{0 < \Delta \leq \frac{1}{N}} A_i$, our assumption leads to a contradiction. So, it must be that $d = d(A, B)$, and thus $d > 0$. This implies the existence of N such that $\forall i \geq N : d_i > 0$. Hence, for such i we have $A_{\Delta_i} \cap B = \varnothing$ (*cf.* Lemma 22). ∎

B Proof of Theorem 14

Lemma 24 ([Pur98, Lemma 7.1]) *Let $p = p_0\, p_1\, \dots\, p_N$ be a (not necessarily cyclic) path in the region graph G, let $\pi = (q_0, t_0)\,(q_1, t_1) \cdots (q_N, t_N)$ and $\pi' = (q'_0, t'_0)\,(q'_1, t'_1) \cdots (q'_N, t'_N)$ be two trajectories s.t. q_i and q'_i belong to p_i for all i. Then for all $\lambda \in [0, 1]$, there exists a trajectory from $\lambda q_0 + (1 - \lambda)q'_0$ to $\lambda q_N + (1 - \lambda)q'_N$ in $[\![A]\!]$.*

Proof. This is immediate by considering the path

$$\pi'' = (\lambda q_0 + (1-\lambda)q'_0, \lambda t_0 + (1-\lambda)t'_0)\,(\lambda q_1 + (1-\lambda)q'_1, \lambda t_1 + (1-\lambda)t'_1) \cdots \\ (\lambda q_N + (1-\lambda)q'_N, \lambda t_N + (1-\lambda)t'_N).$$

This is a trajectory in $[\![A]\!]$ since guards are convex. ■

Lemma 25 ([Pur98, Lemma 7.3]) *Let p be a cycle in the region graph. Then L_p is convex.*

Proof. Let $x, y \in L_p$, and $\lambda \in [0, 1]$. There exists l and m s.t. $x \in L_{k,p}$ and $y \in L_{l,p}$. Then both x and y are in $L_{kl,p}$, and $\lambda x + (1-\lambda)y \in L_{kl,p} \subseteq L$ according to Lemma 24. ■

Definition 26 Let p be a (closed) region. The set of *vertices* of p is the smallest set of points $S(p) = \{v_0, v_1, \dots, v_k\}$ s.t. $p = \mathrm{Conv}(\{v_0, v_1, \dots, v_k\})$. □

This definition is meaningful since regions are polytopes. Such a set is unique, and the number of vertices is bounded by $n+1$, where n is the number of clocks.

Lemma 27 *The vertices of a region p are the valuations in p whose values are integers: $S(p) = p \cap \mathbb{N}^n$*

Proof. Let $v \in p$, and v_0 the valuation s.t. $v_0(x) = \lfloor v(x) \rfloor$. That valuation belongs to p, proving that any region always contains at least one vertex.

There exists sets of clocks X_0, X_1, …, X_k s.t. all valuations w verifying the following equations is in p:

$$\begin{aligned} \forall i.\ \forall x, y \in X_i.\ \langle w(x) \rangle &= \langle w(y) \rangle \\ \forall i < j.\ \forall x \in X_i,\ y \in X_j.\ \langle w(x) \rangle &\leq \langle w(y) \rangle \\ \forall x \in X_0.\ \langle w(x) \rangle &= 0 \end{aligned}$$

Thus $v_0 \in p$ and, for all $0 < j \leq k$, valuation v_j defined as

$$\begin{aligned} v_j(x) &= v_0(x) && \text{if } x \in X_i \text{ with } i < j \\ v_j(x) &= v_0(x) + 1 && \text{if } x \in X_i \text{ with } i \geq j \end{aligned}$$

are in p.

Now let w be a valuation in p. We define

$$w' = (1 - \langle w(x_k)\rangle)v_0 + (\langle w(x_1)\rangle - \langle w(x_0)\rangle)v_1 + \cdots + (\langle w(x_k)\rangle - \langle w(x_{k-1})\rangle)v_k$$

where each x_i is a clock in the corresponding X_i, and claim that w' equals w. Indeed, let y be a clock, and j s.t. $y \in X_j$. Then

$$\begin{aligned} w'(y) &= (1 - \langle w(x_k)\rangle)v_0(y) + (\langle w(x_1)\rangle - \langle w(x_0)\rangle)v_1(y) + \cdots + \\ &\quad (\langle w(x_k)\rangle - \langle w(x_{k-1})\rangle)v_k(y) \\ &= (1 - \langle w(x_k)\rangle)v_0(y) + (\langle w(x_1)\rangle - \langle w(x_0)\rangle)(v_0(y)+1) + \cdots + \\ &\quad (\langle w(x_j)\rangle - \langle w(x_{j-1})\rangle)(v_0(y)+1) + \\ &\quad (\langle w(x_{j+1})\rangle - \langle w(x_j)\rangle)(v_0(y)) + \cdots + \\ &\quad (\langle w(x_k)\rangle - \langle w(x_{k-1})\rangle)(v_0(y)) \\ &= v_0(y) + w(x_j) = w(y) \end{aligned}$$

since $y \in X_j$. Thus $p = \mathrm{Conv}(v_i)$. On the other hand, $\{v_i\}$ is the smallest set generating p, since no such valuation is a combination of the other ones. ■

This ensures that if a region r' is a sub-region of a region r (remember that we consider "closed" regions), then its set of vertices $S(r')$ is the intersection of r' and of the set $S(r)$ of vertices of r.

Definition 28 [see [Pur98, section 7.4]] Let $p = p_0\, p_1\, \ldots\, p_N$ be a cycle in the region graph. The *orbit graph* is the graph $\Theta = (V_\Theta, \rightarrow_\Theta)$ s.t. $V_\Theta = S(p_0)$ and, for all $v, w \in V_\Theta$, $v \rightarrow_\Theta w$ iff $w \in R(v)$.

For $m \in \mathbb{N}$ and $v \in V_\Theta$, we write $\mathsf{Succ}^m(v) = \{w \in V_\Theta \mid v \rightarrow^m_\Theta w\}$, and $\mathsf{Pred}^m(v) = \{w \in V_\Theta \mid w \rightarrow^m_\Theta v\}$. We also define $V = \{v \in V_\Theta \mid \exists m \in \mathbb{N}.\ v \in \mathsf{Succ}^m(v)\}$. □

By Lemma 25, we have $\mathrm{Conv}(V) \subseteq L$.

Lemma 29 ([Pur98, Lemma 7.4]) *Suppose there exists a run in the region graph from p to p'. Then for each vertex v of p, there exists a vertex v' of p' s.t. there is a trajecectory from v to v', and conversely, for each vertex v' of p', there exists a vertex of p s.t. there is a trajectory from v to v'.*

Proof. Let v be a vertex of p. Then $\{v\}$ forms a sub-region of p, and its successors in p' form a subregion of p'. According to the remark following the proof of Lemma 27, that subregion of p' necessarily contains a vertex.

The same argument can be applied backwards, since the predecessor of a subregion of p' is a subregion of p. ■

This entails that any vertex has an outgoing edge in Θ. Thus for all $v \in V_\Theta$, there exists an integer m s.t. $\mathsf{Succ}^m(v) \cap V \neq \varnothing$, because V_Θ is finite. Let M be the largest such m. Then $\mathsf{Succ}^M(v) \cap V \neq \varnothing$ for all v. A similar argument proves the existence of M' s.t. $\mathsf{Pred}^{M'}(v) \cap V \neq \varnothing$ for all v.

Theorem 14 ([Pur98, Lemma 7.10]) *Let $p = p_0\, p_1\, \dots\, p_N$ be a cycle in G. Then for any $z \in p_0$, there exists $z', z'' \in L$ s.t. there exist trajectories in $[\![A]\!]$ from z to z' and from z'' to z.*

Proof. We write $z = \sum_i \lambda_i\, v_i$, where $\lambda_i \in [0, 1]$, $\sum_i \lambda_i = 1$ and $v_i \in V_\Theta$. For each v_i, let w_i be an item of $\mathsf{Succ}^M(v_i) \cap V$. From Lemma 24, there is a path from z to $z' = \sum_i \lambda_i w_i \in \mathrm{Conv}(V) \subseteq L$.

Conversely, if x_i is a vertex in $\mathsf{Pred}^{M'}(v_i) \cap V$, there is a path from $z'' = \sum_i \lambda_i\, x_i \in \mathrm{Conv}(V) \subseteq L$ to z. ■

C Proof of Lemma 15

Lemma 15 *Let A be a timed automaton with n clocks, $\Delta \in \mathbb{Q}^{>0}$, and $\delta = \frac{\Delta}{n}$. Let $p = p_0\, p_1\, p_2 \dots p_N$ be a progress cycle in the region graph. Let u be a valuation in p_0 having a limit cycle, i.e. for which there exists a trajectory $\pi[0, T]$ in $[\![A]\!]_0^0$ following p and with $\pi(0) = \pi(T) = u$. Let $v \in p_0 \cap B(u, \delta)$ be a neighbor valuation. Then there exists a trajectory from u to v in $[\![A]\!]_\Delta^0$.*

Proof. We first define our notations: We assume π is a cycle on valuation u. We write[2]

$$\pi = (l_0, u_0) \xrightarrow{t_0} (l_0, u'_0) \xrightarrow[R_0]{a_0} (l_1, u_1) \xrightarrow{t_1} (l_1, u'_1) \xrightarrow[R_1]{a_1} \cdots$$
$$\cdots \xrightarrow[R_{m-2}]{a_{m-2}} (l_{m-1}, u_{m-1}) \xrightarrow{t_{m-1}} (l_{m-1}, u'_{m-1}) \xrightarrow[R_{m-1}]{a_{m-1}} (l_m, u_m)$$

with $u_0 = u_m = u$. In this run, t_i represents the "duration" of a time transition, a_i represents an action transition along which clocks appearing in R_i are reset. We have in particular $T = \sum_{i=0}^{m-1} t_i$.

We define sets $X_0, \dots, X_q$ of indices s.t., for any valuation w in region p,

- for all $i \in X_0$, $\langle w(i) \rangle = 0$;
- for all $i, j \in X_k$, $\langle w(i) \rangle = \langle w(j) \rangle$;
- for all $k < l$ and for all $i \in X_k,\ j \in X_l$, $\langle w(i) \rangle \leq \langle w(j) \rangle$;

For each index i, we define $\alpha_i \in [0; m]$ to be the index of the transition in which clock x_i is reset for the last time along π. Formally, we have

$$u_i \in R_{\alpha_i} \qquad\qquad u_i \notin \bigcup_{\beta > \alpha_i} R_\beta \tag{1}$$

[2] Other cases are possible, namely paths starting with an action transition and/or ending with a delay transition. Those cases could be treated in the same way.

By Assumption 8, we know that each clock is reset at least once along π. Thus, for each clock x_i, we have

$$u_m(i) = u(i) = \sum_{\beta > \alpha_i} t_\beta \tag{2}$$

Now regarding valuation v, we define $\boldsymbol{\delta} = (\delta_i) = (v(i) - u(i))$. By hypothesis, $|\delta_i| \leq \delta$. Moreover, $v \in p_0$ implies that the order of fractional parts of the clocks must be the same in u and v:

- for all $i \in X_0$, $\langle v(i) \rangle = 0$;
- for all $i, j \in X_k$, $\langle v(i) \rangle = \langle v(j) \rangle$;
- for all $k < l$ and for all $i \in X_k,\ j \in X_l$, $\langle v(i) \rangle \leq \langle v(j) \rangle$;

This entails that

- for all $i, j \in X_k$, $\delta_i = \delta_j$;
- for any i, j s.t. $u(i) < u(j)$ (and hence $\alpha_i > \alpha_j$ since (2)), we have $u(i) + \delta_i \leq u(j) + \delta_j$, and

$$\delta_i - \delta_j \leq \sum_{\beta = \alpha_j + 1}^{\alpha_i} t_\beta \tag{3}$$

We now build the run π' between u and u': For each $i \in [0; m]$, we define

$$t'_i = \begin{cases} t_i - \delta_j & \text{if } i = \alpha_j \\ t_i & \text{otherwise} \end{cases} \qquad\qquad t''_{i+1} = \begin{cases} t'_{i+1} + \delta_j & \text{if } i = \alpha_j \\ t'_{i+1} & \text{otherwise} \end{cases} \tag{4}$$

For the sake of simplicity, we temporarily assume that each t''_i is positive, that $\alpha_j \in [1; m-2]$ and $\alpha_j \neq \alpha_{j'} + 1$ for any j, j', and define the run π' from π by replacing t_i with t''_i:

$$\pi' = (l_0, v_0) \xrightarrow{t''_0} (l_0, v'_0) \xrightarrow[R_0]{a_0} (l_1, v_1) \xrightarrow{t''_1} (l_1, v'_1) \xrightarrow[R_1]{a_1} \cdots$$
$$\cdots \xrightarrow[R_{m-2}]{a_{m-2}} (l_{m-1}, v_{m-1}) \xrightarrow{t''_{m-1}} (l_{m-1}, v'_{m-1}) \xrightarrow[R_{m-1}]{a_{m-1}} (l_m, v_m)$$

with $v_0 = u_0 = u$. We claim that π' corresponds to a trajectory in $[\![A]\!]^0_\Delta$. Indeed, the difference $|u(i) - v(i)|$ is always bounded with $n\delta$: The only difference between π and π' is that some clocks may sometimes be reset a little bit earlier or later, resulting in shifting their value with at most δ. This can only occur once for each clock, *i.e.* at most n times, thus the global difference between $u(i)$ and $v(i)$ is at most Δ. This ensures that each action transition can be fired along π' in $[\![A]\!]^0_\Delta$ since they were firable along π in $[\![A]\!]$.

Since clocks are reset at exactly the same transitions as along π, equation (2) applies here, and we get

$$v_m(j) \;=\; \sum_{\beta > \alpha_j} t''_\beta \;=\; t''_{\alpha_j + 1} + \sum_{\beta > \alpha_j + 1} t''_\beta \;=\; t_{\alpha_j + 1} + \delta_j + \sum_{\beta > \alpha_j + 1} t''_\beta$$

Since $t''_k + t''_{k+1} = t_k + t_{k+1}$ when k equals some α_l, and $t_k = t''_k$ when no α_l appears in $\{k, k+1\}$, we get

$$v_m(j) \;=\; \delta_j + \sum_{\beta > \alpha_j} t_\beta \;=\; u_m(j) + \delta_j \tag{5}$$

Thus $v_m = v$. It follows that π' witnesses the fact that $u \to v$ in $[\![A]\!]^0_\Delta$.

Let's consider "border cases" now:

- If $\alpha_k = \alpha_j + 1$ for some j and k, then t''_{α_j+1} equals $t_{\alpha_j+1} - \delta_k + \delta_j$. However, this does not change our result, since we still get equation (5);
- If $\alpha_j = 0$ for some j and π starts with an action transition, then we have $t''_1 = t_1 + \delta_j$ (or $t_1 + \delta_j - \delta_k$), which "enlarges"[3] the total duration of the path by δ_j. Since in that case the final value of clock x_j is the total duration of the path (because x_j is only reset at the very beginning of π), this also enlarges the final value of x_j by δ_j at the end of π', exactly as expected;
- Symmetrically, if $\alpha_j = m-1$ for some j and π ends with an action transition, then clock x_j equals zero in u, *i.e.* clock $x_j = 0$ in p_0, and p_0 is a boundary region. Necessarily, $\delta_j = 0$, and we don't have to care about it.
- It could happen that $t''_i < 0$ for some i. However, it should be noticed that the small "shifts" that are applied to only one transition in the construction above, could be distributed over all time transitions occurring between transitions t_{α_j+1} and t_{α_k}, for "consecutive" α_j and α_k. Equation (3) ensures that we can distribute $\delta_j - \delta_k$ and get positive durations t''_i. ■

D Proofs of section 6.3

Lemma 19 *Suppose $r \to_G r'$ in the region graph G of a timed automaton, and $u \xrightarrow{\tau} v$ with $u \in r$, $v \in r'$ and $\tau \in \mathbb{R}^{\geq 0}$. Then for any $x \in B(u, R) \cap r$, there exists $y \in B(v, 2R) \cap r'$ such that $x \xrightarrow{\tau'} y$ for some τ' with $|\tau - \tau'| \leq R$.*

Proof. We assume that regions r and r' are defined by the following inequalities:

$$u_i - u_j \leq m_{i,j} \qquad\qquad \alpha_i \leq u_i \leq \beta_i \tag{r}$$
$$v_i - v_j \leq m'_{i,j} \qquad\qquad \alpha'_i \leq v_i \leq \beta'_i \tag{r'}$$

Also, we assume that $m_{i,j} \leq \beta_i - \alpha_j$ and $m'_{i,j} \leq \beta'_i - \alpha'_j$. This entails, in particular, that

$$u_i - u_j \leq \beta_i - \alpha_j \qquad\qquad v_i - v_j \leq \beta'_i - \alpha'_j \tag{6}$$

[3] Remember that δ_j could be negative.

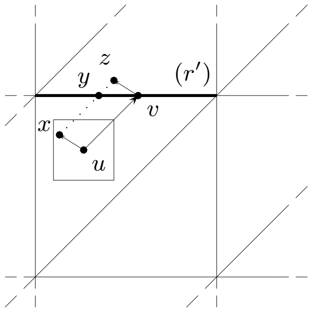
Moreover, since $v = u + \tau$, we have

$$m_{i,j} = m'_{i,j} \qquad\qquad v_i - v_j = (u_i + \tau) - (u_j + \tau) \leq \beta_i - \alpha_j \tag{7}$$

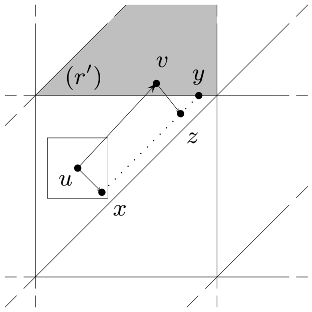
We define $\boldsymbol{\delta} = x - u$. Note that $\|\boldsymbol{\delta}\| \leq R$. Since $x \in r$, we have

$$(u_i + \delta_i) - (u_j + \delta_j) \leq m_{i,j} \qquad\qquad \alpha_i \leq u_i + \delta_i \leq \beta_i \tag{8}$$

Now define $z = v + \boldsymbol{\delta}$. As can be seen on Fig. 6, z might not belong to r', and we have to find a neighbor y of z that belongs to r' and s.t. $x \to y$.



(a) If the interior of r' is empty



(b) If the interior of r' is not empty

Fig. 6. Building a successor y of x in region r'.

If, for some i_0, $\alpha'_{i_0} = \beta'_{i_0}$*:* We define $t = -\delta_{i_0}$. The value of t does not depend on the choice of i_0. Indeed, if for some $j \neq i_0$, we have $\alpha'_j = \beta'_j$, then

$$v_{i_0} - v_j = \alpha'_{i_0} - \alpha'_j = m'_{i_0,j} = -m'_{j,i_0}$$

since the values of v_{i_0} and v_j are fixed in region r'. This entails that

$$u_{i_0} - u_j = m_{i_0,j} = -m_{j,i_0} = (u_{i_0} - \delta_{i_0}) - (u_j - \delta_j)$$

since diagonal constraints are the same in r and r'. The previous equalities give $\delta_{i_0} = \delta_j$, as claimed above.

We now define $y = z + t$. Obviously

$$y_i - y_j = (v_i + \delta_i + t) - (v_j + \delta_j + t) = x_i - x_j \leq m'_{i,j}.$$

Moreover:

– For any j s.t. $\alpha'_j = \beta'_j$, we have

$$y_j = v_j + \delta_j + t = v_j + \delta_j - \delta_j = v_j = \alpha'_j = \beta'_j$$

since $t = -\delta_j$;

– For any j s.t. $\alpha'_j \neq \beta'_j$, we have

$$\begin{aligned} y_j - y_{i_0} &= (v_j + \delta_j + t) - (v_{i_0} + \delta_{i_0} + t) \\ &= (v_j + \delta_j) - (v_{i_0} + \delta_{i_0}) \leq m'_{j,i_0}. \end{aligned}$$

Since $y_{i_0} = \alpha'_{i_0}$, we get $y_j \leq m'_{j,i_0} + \alpha'_{i_0} \leq \beta'_j$. In the same way, we have

$$\begin{aligned} y_{i_0} - y_j &= (v_{i_0} + \delta_{i_0} + t) - (v_j + \delta_j + t) \\ &= (v_{i_0} + \delta_{i_0}) - (v_j + \delta_j) \leq m'_{i_0,j} \end{aligned}$$

and we obtain $\alpha'_j \leq y_j$.

Since $|t| \leq R$, we have $\|y - v\| \leq 2R$ and $\tau' = \tau + t$ so that $|\tau - \tau'| \leq R$, which concludes the proof for this subcase.

If, for all i, $\alpha'_i < \beta'_i$: We define two sets $I = \{i_0, \ldots, i_p\}$ and $I' = \{i'_0, \ldots, i'_q\}$ s.t.

$$\forall i \in I.\ \alpha'_i > z_i \qquad\qquad \forall i' \in I'.\ z_{i'} > \beta'_{i'}$$

If both I and I' are empty, then z belongs to r' and the proof is over (diagonal constraints are satisfied thanks to the first equation of (8)).

Otherwise, assume I is not empty, and define $t = \max\{\alpha'_i - z_i \mid i \in I\}$. We write i_0 for the index s.t. $t = \alpha'_{i_0} - z_{i_0}$. Obviously, in this case, $t > 0$. Moreover, $t = \alpha'_{i_0} - v_{i_0} - \delta_{i_0}$, and since $\alpha'_{i_0} \leq v_{i_0}$, we have $t \leq -\delta_{i_0}$.

We now define $y = z + t$. Diagonal constraints are still obviously verified by y. Moreover, for $i \in I$, $\alpha_i \leq z_i + t = y_i$ by construction of t. For $j \notin I$, $\alpha'_j \leq z_j < z_j + t = y_j$ since $0 < t$.

Now assume that, for some index j, we have $y_j > \beta'_j$. Then $y_j - y_{i_0} > \beta'_j - \alpha'_{i_0}$, which is impossible since $y_j - y_{i_0} = x_j - x_{i_0} \leq m'_{j,i_0} \leq \beta'_j - \alpha'_{i_0}$. This proves that $y \in r'$. Since $0 < t \leq -\delta_{i_0} \leq R$, we have $\|y - v\| \leq 2R$ and $|\tau - \tau'| \leq R$.

The case where I is empty (and I' is not) is similar: Define $t = \min\{\beta'_{i'} - z_{i'} \mid i' \in I'\}$, and write i'_0 for the index s.t. $t = \beta'_{i'_0} - z_{i'_0}$. This time, $t < 0$ Moreover, $t = \beta_{i'_0} - u_{i'_0} - \delta_{i'_0}$, and since $\beta'_{i'_0} - v_{i'_0} \geq 0$, we have $-\delta_{i'_0} \leq t < 0$.

We define $y = z + t$. By construction of t, for $i' \in I'$, we have $y_{i'} \leq \beta'_{i'}$. For $j \notin I'$, we have $y_j = z_j + t < z_j \leq \beta'_j$ since $t < 0$.

Assume that, for some j, $\alpha'_j > y_j$. Then $y_{i'_0} - y_j < \beta'_{i'_0} - \alpha'_j$, which is a contradiction since $y_{i'_0} - y_j = x_{i'_0} - x_j \leq m'_{i'_0,j} \leq \beta'_{i'_0} - \alpha'_j$. As in the previous case, this proves that $y \in r'$, $\|y - v\| \leq 2R$ and $|\tau - \tau'| \leq R$ since $-R \leq -\delta_{i'_0} \leq t < 0$. ∎

Lemma 30 *Let $x \in \mathbb{R}^n$ and $\rho \in \mathbb{R}^{\geq 0}$. Let $D_i = \{z \in \mathbb{R}^n \mid z_i \in \mathbb{Z}\}$.*

(a) If $B(x,\rho) \cap D_i \neq \varnothing$ and $B(x,\rho) \cap D_j \neq \varnothing$, then $B(x,\rho) \cap D_i \cap D_j \neq \varnothing$.
(b) If $\rho < \frac{1}{2}$ and $\exists z \in B(x,\rho) \cap D_i$ then $\forall z' \in B(x,\rho) \cap D_i : z_i = z'_i$.

Lemma 31 *For any $a, b, \epsilon \in \mathbb{R}$,*

(a) If $|a| \leq \epsilon$, then $\langle a \rangle \leq \epsilon$ or $\langle a \rangle \geq 1 - \epsilon$.
(b) If $\langle a - b \rangle \leq \epsilon$, then $|\langle a \rangle - \langle b \rangle| \leq \epsilon$ or $|\langle a \rangle - \langle b \rangle| \geq 1 - \epsilon$.
(c) If $\langle a - b \rangle \geq 1 - \epsilon$, then $|\langle a \rangle - \langle b \rangle| \leq \epsilon$ or $|\langle a \rangle - \langle b \rangle| \geq 1 - \epsilon$.

Theorem 18 *For any distance $\delta \in \mathbb{R}^{>0}$, for any number of steps $k \in \mathbb{N}$, there exists $\Delta \in \mathbb{Q}^{>0}$ such that for any k-step trajectory $\pi' = (q'_0, t'_0) \ldots (q'_k, t'_k)$ of $[\![A]\!]^0_\Delta$, there exists a k-step trajectory $\pi = (q_0, t_0) \ldots (q_k, t_k)$ of $[\![A]\!]$ such that for any position i $(0 \leq i \leq k)$*

- *$q_i \in [q'_i]$, that is the two trajectories cross the same regions,*
- *and the distance between corresponding states q_i and q'_i is bounded by δ, that is $q_i = (l_i, v_i)$, $q'_i = (l'_i, v'_i)$ where $l_i = l'_i$ and $\|v_i - v'_i\| \leq \delta$.*

Proof of theorem 18. We assume the existence of Δ, and we collect sufficient conditions along the proof which will entail the result. We will be using the following symbols: $r = 2\Delta$, $\eta = \Delta$. Furthermore, we make the non restrictive hypothesis that $\delta < \frac{1}{2}$, and we assume $nr < \frac{1}{2}$, that is $\Delta < \frac{1}{4n}$.

The outline of the proof is as follows. We define a "simulation" relation $\mathcal{R}$ between valuations, which is a subset of $\mathbb{R}^n \times \mathbb{N} \times \mathbb{R}^n$ and we write $\mathcal{R}_a(x', x)$ if $(x', a, x) \in \mathcal{R}$. We show that

$$\text{If } \begin{cases} a \leq k \\ \mathcal{R}_a(x', x) \\ x' \xrightarrow{t'} y' \xrightarrow{\sigma} z' \text{ in } [\![A]\!]^0_\Delta \end{cases} \quad \text{then } \exists y, z : \begin{cases} x \xrightarrow{t} y \xrightarrow{\sigma} z \text{ in } [\![A]\!] \\ \mathcal{R}_{a+1}(z', z) \\ \|z - z'\| \leq \|y - y'\| \leq \delta \end{cases}$$

The relation $\mathcal{R}_a$ takes into account the current position in the trajectory: if x' is the a^{th} state in the trajectory, then x is in a neighbourhood of size parameterized by a. This size increases with a, because the longer is the trajectory, the greater is the possible deviation (Fig. 4). This is not a classical simulation relation because it only works for a fixed trajectory length k (which is sufficient for proving Theorem 18).

We define a distance $\mathcal{D} : [0, 1[\times [0, 1[\to [0, 1[$ between fractional parts of clocks as follows: seeing the segment $[0, 1[$ as a circle joining the points 0 and 1, $\mathcal{D}(a, b)$ is the shortest distance between a and b on this circle. Formally, $\mathcal{D}(a, b) = \min(|a - b|, 1 - |a - b|)$.

Intuitively, $\mathcal{R}$ links a valuation x with a valutation x' if whenever two clock have *close* fractional parts in x' (in the sense of $\mathcal{D}$), those clock have *identical* fractional parts in x (Fig. 5).

Definition of $\mathcal{R}$. Let $x' = (x'_1, \ldots, x'_n)$. We define a relation $\sim \subseteq I \times I$ where $I = \{1, \ldots, n\}$:

$$i \sim j \text{ iff } \mathcal{D}(\langle x'_i \rangle, \langle x'_j \rangle) \leq r$$

Consider $\sim^*$, the reflex-transitive closure of $\sim$. Let J be an equivalence class of $\sim^*$. Define $m_J = \min_{i \in J}\{\langle x'_i \rangle\}$ and $M_J = \max_{i \in J}\{\langle x'_i \rangle\}$. Note that the set $\{\langle x_i \rangle \mid i \in J\}$ can have two different forms (see J and K in Fig. 7), depending on whether $M_J - m_J < \frac{1}{2}$ or not (with the assumption that $nr < \frac{1}{2}$).

For having $\mathcal{R}_a(x', x)$, we want x_i and x_j to have identical fractional parts ($\langle x_i \rangle = \langle x_j \rangle$) if $i, j \in J$. For clocks in equivalence classes of the form of J in Fig. 7, we choose a unique value $val_a(J)$ of the fractional parts in the interval $[m_J - \theta_a, M_J + \theta_a]$ (where θ will be precisely defined later, and must satisfy the system (S) of constraints given at the end of this proof). Since such intervals, can overlap for different classes, we also require that the order on fractional part is preserved by asking that $x \in [x']$. For classes of the form of K in Fig. 7, we distinguish clocks with fractional parts close to 0 (which are rounded to 0) and clocks with fractional parts close to 1 (which are rounded to 1), but all will have a fractional part equal to 0, as required.

Formally, we define $\mathcal{R}$ as follows:

$$\mathcal{R}_a(x', x) \text{ iff } x \in [x'] \text{ and } \forall i \in I : x_i = \lfloor x'_i \rfloor + f_i$$

where, for $j \in J$,

$$f_j = \begin{cases} val_a(J) & \text{if } M_J - m_J < \frac{1}{2} \\ 0 & \text{if } M_J - m_J > \frac{1}{2} \text{ and } \langle x'_j \rangle < \frac{1}{2} \\ 1 & \text{if } M_J - m_J > \frac{1}{2} \text{ and } \langle x'_j \rangle > \frac{1}{2} \end{cases}$$

and for J such that $M_J - m_J < \frac{1}{2}$, $val_a(J)$ is chosen in the interval $[m_J - \theta_a, M_J + \theta_a]$ with the restriction that if this interval contains an integer (0 or 1, but never both), then that integer value *must* be chosen. The value θ_a is defined by a system (S) of constraints given at the end of this proof.

We can show that given x', there always exists x such that $\mathcal{R}_a(x, x')$. It suffices to take $val_a(J) \in [m_J, M_J]$ when $M_J - m_J < \frac{1}{2}$ and to see that the order of the fractional part of x is the same as the order of the fractional part of x', except that some strict inequalities are replaced by non-strict inequalities; this is not problematic since the $[x']$ is the closure of $]x'[$.

Properties of $\mathcal{R}$. We give on one hand a necessary condition, and on the other hand a sufficient condition to have $\mathcal{R}_a(x', x)$.

It should be clear to the reader that

$$\text{If } \mathcal{R}_a(x', x), \text{ then } \begin{cases} (1)\ x \in [x'] \\ (2)\ d(x, x') \leq nr + \theta_a \\ (3)\ \mathcal{D}(\langle x'_i \rangle, \langle x'_j \rangle) \leq r \rightarrow \langle x_i \rangle = \langle x_j \rangle \\ (4)\ (\exists z \in \mathbb{Z} : |z - x'_i| \leq \frac{r}{2}) \rightarrow x_i = z \end{cases}$$

Consition (4) is a consequence of constraint (S1): $\theta_a \geq \frac{r}{2}$.

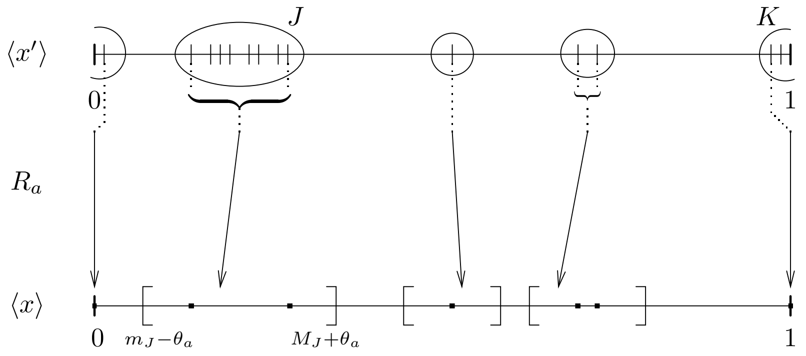


Fig. 7. Fractional parts of x wrt fractional parts of x' when $\mathcal{R}_a(x', x)$. J and K are equivalence classes of $\sim^*$.

The following sufficient condition is more tricky

$$\text{If } \left\{ \begin{array}{l} (1')\ x \in [x'] \\ (2')\ d(x, x') \leq \theta_a \\ (3')\ \mathcal{D}(\langle x'_i \rangle, \langle x'_j \rangle) \leq r \to \langle x_i \rangle = \langle x_j \rangle \\ (4')\ (\langle x'_i \rangle \leq \frac{r}{2} \vee \langle x'_i \rangle \geq 1 - \frac{r}{2}) \to \langle x_i \rangle = 0 \end{array} \right\} \text{ then } \mathcal{R}_a(x', x).$$

Condition $(3')$ ensures that clocks with "close" fractional parts are mapped to clocks with identical fractional parts. That condition, in combination with condition $(4')$ implies that if two clocks i and j have fractional parts such that $|\langle x'_i \rangle - \langle x'_j \rangle| \geq 1 - r$ then $\langle x'_i \rangle = \langle x'_j \rangle = 0$ (which is necessary to lie in $\mathcal{R}$). Condition $\theta_a \geq \frac{r}{2}$ (S1) ensures that we really *can* choose to take $\langle x_i \rangle = 0$ when $(4')$ is satisfied. Finally, with $(1')$ and $(2')$, we get a sufficient condition: let us show that if J is an equivalence class of $\sim^*$ such that $M_J - m_J < \frac{1}{2}$ and $i \in J$, then $val_a(J) \in [m_J - \theta_a, M_J + \theta_a]$. Let $j_0 \in J$ be such that $|\langle x_i \rangle - \langle x'_{j_0} \rangle| = \max_{j \in J}\{|\langle x_i \rangle - \langle x'_j \rangle|\}$. Then, for any $j \in J$ we have

$$|val_a(J) - \langle x'_j \rangle| = |\langle x_i \rangle - \langle x'_j \rangle| \leq |\langle x_{j_0} \rangle - \langle x'_{j_0} \rangle| \leq \theta_a$$

Then $\langle x'_{j_0} \rangle - \theta_a \leq val_a(J) \leq \langle x'_{j_0} \rangle + \theta_a$, and since $\langle x'_{j_0} \rangle \in [m_J, M_J]$, we obtain the claimed result.

Simulation.

A. We show that if $x' \xrightarrow{t'} y' \xrightarrow{\sigma} z'$ in $[\![A]\!]^0_\Delta$ (that is $y' = x' + t'$, $y' \in [\![g]\!]_\Delta$ and $z' = y'[X := 0]$) and $\mathcal{R}_a(x', x)$, then we can determine t such that $x \xrightarrow{t} y \xrightarrow{\sigma} z$ in $[\![A]\!]$ (that is $y = x + t$, $y \in [\![g]\!]$ and $z = y[X := 0]$) and $\mathcal{R}_{a+1}(z', z)$.

We consider two cases:

1. If $\exists i \in I, \gamma \in B(y', \eta) : \gamma_i \in \mathbb{Z}$

Then, for such i, define $t = \gamma_i - x_i$. We have

$$\begin{aligned}|t - t'| = |(\gamma_i - x_i) - (y'_i - x'_i)| &\leq |(\gamma_i - y'_i) - (x_i - x'_i)| \\ &\leq |\gamma_i - y'_i| + |x_i - x'_i| \\ &\leq \eta + nr + \theta_a\end{aligned}$$

and similarly,

$$\begin{aligned}|y - y'| = |(x + t) - (x' + t')| &\leq |(x - x') - (t - t')| \\ &\leq |x - x'| + |t - t'| \\ &\leq \eta + 2nr + 2\theta_a \leq \delta \quad \text{(S2)}\end{aligned}$$

The value of t does not depend on the choice of i. Indeed, if for some j, there exists $\gamma' \in B(y', \eta) : \gamma'_j \in \mathbb{Z}$, then by Lemma 30(a) there exists $\gamma \in B(y', \eta)$: $\gamma_i \in \mathbb{Z} \wedge \gamma_j \in \mathbb{Z}$. Hence, for such γ we have $|y'_i - \gamma_i| \leq \eta$ and $|y'_j - \gamma_j| \leq \eta$. And, successively
$|(y'_i - \gamma_i) - (y'_j - \gamma_j)| \leq |y'_i - \gamma_i| + |y'_j - \gamma_j| \leq 2\eta$
$|(y'_i - y'_j) - (\gamma_i - \gamma_j)| \leq 2\eta$
$|(x'_i - x'_j) - (\gamma_i - \gamma_j)| \leq 2\eta \quad$ (since $y' = x' + t'$)
$\langle x'_i - x'_j \rangle \leq 2\eta \vee \langle x'_i - x'_j \rangle \geq 1 - 2\eta \quad$ (Lemma 31(a) and $\langle \gamma_i - \gamma_j \rangle = 0$)
$|\langle x'_i \rangle - \langle x'_j \rangle| \leq 2\eta \vee |\langle x'_i \rangle - \langle x'_j \rangle| \geq 1 - 2\eta \quad$ (Lemma 31(b) and 31(c))
$|\langle x'_i \rangle - \langle x'_j \rangle| \leq r \vee |\langle x'_i \rangle - \langle x'_j \rangle| \geq 1 - r \quad (r = 2\eta)$
$\langle x_i \rangle = \langle x_j \rangle \quad$ (we have $\mathcal{R}_a(x', x)$ which implies equation (3))
Hence, we have $y_j = x_j + t \in \mathbb{Z}$. Since $|y - y'| \leq \eta + 2nr + 2\theta_a < \frac{1}{2}$ (S3), from Lemma 30(b) it must be that $y_j = \gamma_j$ and we have $t = \gamma_j - x_j$.
Note that we must show that $t \geq 0$, that is $x_i \leq \gamma_i$. If $x'_i \leq \gamma_i$, then since $x \in [x']$ and $\gamma_i \in \mathbb{Z}$, we have $x_i \leq \gamma_i$. If $x'_i \geq \gamma_i$, then since $x'_i \leq y'_i$, we have $|x'_i - \gamma_i| \leq \eta = \frac{r}{2}$. Consequently, from equation (4), $x_i = \gamma_i$ and $t = 0$.

2. If $\forall i \in I, \gamma \in B(y', \eta) : \gamma_i \notin \mathbb{Z}$
Since $x \in B(x', nr + \theta_a)$ and $x \in [x']$, from Lemma 19 there exists $y \in B(y', 2nr + 2\theta_a) \cap [y']$ such that $x \xrightarrow{t} y$ with $|t - t'| \leq nr + \theta_a$. Note that the bounds we had on $|t - t'|$ and $|y - y'|$ in the previous case are still satisfied in this case.

In both cases, we have $\mathcal{R}_{a+1}(y', y)$: it should be clear to the reader that condition $1'$, $3'$ and $4'$ (since $\eta = \frac{r}{2}$) are satisfied. Condition $2'$ is true if $\eta + 2nr + 2\theta_a = \theta_{a+1}$ (S4).

B. Now, we show that if $y' \in [\![g]\!]_\Delta$ then $y \in [\![g]\!]$.

1. If $B(y', \eta) \cap D_i \neq \varnothing$, then let $\gamma \in B(y', \eta) \cap D_i$. We have $\gamma_i = y_i \in \mathbb{Z}$. Since $|y_i - y'_i| \leq \eta + 2nr + 2\theta_a$ and $\eta + 2nr + 2\theta_a + \Delta < 1$ (S3), the fact that $a_i - \Delta \leq y'_i \leq b_i + \Delta$ implies that $a_i - 1 < y_i < b_i + 1$ which entails that $a_i \leq y_i \leq b_i$ since $y_i \in \mathbb{Z}$.

2. If $B(y', \eta) \cap D_i = \varnothing$, then if $a_i - \Delta \leq y'_i \leq b_i + \Delta$, it is impossible that $a_i - \Delta \leq y'_i \leq a_i$ because it was so, the point $(y_1, \ldots, y_{i-1}, a_i, y_{i+1}, \ldots y_n)$ would belong to $B(y', \eta) \cap D_i$ (since $\eta = \Delta$). Similarly, we cannot have $b_i \leq y'_i \leq b_i + \Delta$. Therefore, we have $a_i \leq y'_i \leq b_i$, and since $y \in [y']$, also $a_i \leq y_i \leq b_i$.

C. Finally, we have $\mathcal{R}_{a+1}(z', z)$ and $|z - z'| \leq \delta$ since resetting clocks reinforces condition $2'$ and does not affect the other conditions.

D. Below, we summarize the constraints (S) we have on θ_a, for $0 \leq a \leq k$

$$\theta_a \geq \frac{r}{2} \tag{S1}$$

$$\eta + 2nr + 2\theta_a < \delta \tag{S2}$$

$$\eta + 2nr + 2\theta_a + \Delta < \frac{1}{2} \tag{S3}$$

$$\eta + 2nr + 2\theta_a = \theta_{a+1} \tag{S4}$$

Let $A = \eta + 2nr = \Delta + 4n\Delta$. We can take $\theta_a = (A + \Delta)2^a - A$ which satisfy the first and last constraint. The other two reduce to $(A + \Delta)2^k < \delta < \frac{1}{2} - \Delta$ which is verified if $\Delta < \frac{1}{2} - \delta$ and $\Delta < \frac{\delta}{2^k(2+4n)}$.

Finally, putting the constraints on Δ all together gives

$$\begin{aligned} &\Delta < \frac{1}{4n} \quad \text{from assumption that } nr < \frac{1}{2} \\ &\Delta < \frac{1}{2} - \delta \\ &\Delta < \frac{\delta}{2^k(2+4n)} \end{aligned}$$

which is clearly satisfiable. ∎

Theorem 20 *Suppose $x \in J^*$ and there exists a trajectory from x to y in $[\![A]\!]^0_\Delta$ with $\Delta < \frac{\alpha}{2^{W+1}(2+4n)}$ where $\alpha < \frac{1}{2n}$, and W is the total number of regions. Then $d(y, J^*) < \alpha$.*

Proof of Theorem 20. Suppose $\pi' = (q'_0, t'_0)(q'_1, t'_1) \ldots (q'_m, t'_m)$ is a stutter-free trajectory of $[\![A]\!]^0_\Delta$ where $q'_0 = x$, $q'_m = y$ and $q'_0 \in J^*$. We need to show that $d(q'_m, J^*) < \alpha$. The proof will be by using induction. Using Theorem 18, $d(q'_i, J*) < \alpha$ for $i \leq W+1$. Now assume (induction hypothesis) that $d(q'_i, J*) < \alpha$ for $i \leq N$ with $N \geq W + 1$. We must show that $d(q'_{N+1}, J^*) < \alpha$. By our induction hypothesis $d(q'_j, J^*) < \alpha$ for $j = (N + 1) - (W + 1)$. Consider $p = p_j p_{j+1} \ldots p_{N+1}$ where $p_i = [q'_i]$ $(j \leq i \leq N + 1)$. From Theorem 18, there is a trajectory $\eta = (q_j, t_j)(q_{j+1}, t_{j+1}) \ldots (q_{N+1}, t_{N+1})$ in $[\![A]\!]$ where for $j \leq i \leq N+1$, we have $\|q_i - q_{i'}\| < \alpha$ and $q_i \in [q'_i]$. Now $d(q_j, J^*) \leq \|q_j - q'_j\| + d(q'_j, J^*) <$

$\frac{1}{2n} + \frac{1}{2n} = \frac{1}{n}$. Therefore from Lemma 11 $[q_j] \cap J^* \neq \varnothing$. By construction of J^* (since a reachability computation is performed (line 4 of Algorithm 1) and the path $[q_j][q_{j+1}] \ldots [q_{N+1}]$ contains a cycle), we have $[q_{N+1}] \subseteq J^*$. It follows that $q_{N+1} \in J^*$. Therefore $d(q'_{N+1}, J^*) < \alpha$. ■