Formal Methods in System Design manuscript No.
(will be inserted by the editor)

Robust Safety of Timed Automata

Martin De Wulf · Laurent Doyen ·
Nicolas Markey · Jean-François Raskin

the date of receipt and acceptance should be inserted later

Abstract Timed automata are governed by an idealized semantics that assumes a perfectly precise behavior of the clocks. The traditional semantics is not robust because the slightest perturbation in the timing of actions may lead to completely different behaviors of the automaton. Following several recent works, we consider a relaxation of this semantics, in which guards on transitions are widened by $\Delta > 0$ and clocks can drift by $\varepsilon > 0$. The relaxed semantics encompasses the imprecisions that are inevitably present in an implementation of a timed automaton, due to the finite precision of digital clocks.

We solve the safety verification problem for this robust semantics: given a timed automaton and a set of bad states, our algorithm decides if there exist positive values for the parameters Δ and ε such that the timed automaton never enters the bad states under the relaxed semantics.

1 Introduction

Timed automata. Timed and hybrid systems are dynamical systems with both discrete and continuous components. A paradigmatic example of a hybrid system is a digital

This research was supported by the Belgian FNRS grant 2.4530.02 of the FRFC project "Centre Fédéré en Vérification" and by the project "MoVES", an Interuniversity Attraction Poles Programme of the Belgian Federal Government.

Martin De Wulf
Dépt Informatique, Université Libre de Bruxelles (ULB), Campus de la Plaine, Bd du Triomphe CP 212, 1050 Bruxelles – Belgium

Laurent Doyen
École Polytechnique Fédérale de Lausanne (EPFL), Station 14, 1015 Lausanne – Switzerland

Nicolas Markey
Laboratoire Spécification & Vérification (LSV), ENS Cachan & CNRS, 61 av. Pdt Wilson, 94230 Cachan – France

Jean-François Raskin
Dépt Informatique, Université Libre de Bruxelles (ULB), Campus de la Plaine, Bd du Triomphe CP 212, 1050 Bruxelles – Belgium

embedded control program for an analog plant environment, like a furnace or an airplane: the controller state moves discretely between control modes, and in each control mode, the plant state evolves continuously according to physical laws. Behaviors of controllers for physical systems are often subject to real-time constraints. A natural model for such controllers is the *timed automaton* model introduced by Alur and Dill [AD94]. Timed automata extend finite state automata with continuous variables called *clocks*. Those clocks take their values in the nonnegative real numbers and count time; they can be reset and compared to other clocks or constants in guards labeling edges and invariants labeling states of the automaton. Several verification and control problems have been studied for timed automata, see for example [HNSY92, MPS95, CHR02], and verification tools have been developed, *e.g.* [BDL$^+$06].

When a high-level description of a controller has been proven *correct* it would be valuable to ensure that an implementation of that controller can be obtained in a systematic way in order to ensure the *preservation of correctness*. This is often called program refinement: given a high-level description P_1 of a program, refine that description into another description P_2 such that the "important" properties of P_1 are maintained. Usually, P_2 is obtained from P_1 by reducing non-determinism. To reason about the correctness of P_2 w.r.t. P_1, we often use a notion of simulation [Mil80] which is powerful enough to ensure conservation of LTL properties for example.

Unfortunately, for timed automata 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*, defining 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 timeouts while implementations can only react within a given, usually small but non-zero, reaction delay. Third, timed automata may describe control strategies that are *unrealistic*, such as Zeno-strategies or strategies that require the controller to act faster and faster [CHR02]. For those reasons, a model of a digital controller that has been proven correct in the traditional semantics may not be implementable (at all) or it may not be possible to turn it systematically into an implementation that is still correct. correct w.r.t. this model.

Implementability of timed automata. To overcome those problems, [DDR05] proposed an alternative semantics for timed automata, which takes into account the digital and imprecise aspects of the hardware in which the automaton is being executed. The hardware is assumed to repeatedly execute the following procedure: it first reads the value of the global clock, then evaluates the guard of each transition, and executes one of the enabled transitions. This procedure is assumed to run in at most Δ_L time units. Moreover, the global (digital) clock is updated at least every Δ_P time units, and its value may drift by some value ε (*i.e.*, the delay u between two updates of the clock is at most Δ_P, and the clock is incremented by some value c between $u(1-\varepsilon)$ and $u(1+\varepsilon)$). The resulting set of executions is denoted by $[\![\mathcal{A}]\!]^{\mathsf{Prg}}_{\Delta_L,\Delta_P,\varepsilon}$ and called the *program semantics*. The automaton $\mathcal{A}$ is said to be *implementable* w.r.t. a given property $\mathcal{P}$ iff there exists positive values for the parameters Δ_L, Δ_P and ε for which all the executions of $[\![\mathcal{A}]\!]^{\mathsf{Prg}}_{\Delta_L,\Delta_P,\varepsilon}$ satisfy $\mathcal{P}$. This model of an implementation platform is clearly implementable by an hardware with bounded imprecision, and the semantics of the implementation platform is kept deliberately simple: any platform that ensures the minimal performances that are imposed by the program semantics ensures compliance with the formal model (see [DDR05] for more discussion and details).

In [DDR05], a different semantics is also introduced for the purpose of verification. This new semantics is called the *Almost-ASAP semantics* and is obtained by enlarging the guards of $\mathcal{A}$ by a parameter Δ. allowing the clocks to drift by ε. The set of executions of $\mathcal{A}$ under this semantics is denoted by $[\![\mathcal{A}]\!]^{\mathsf{AAsap}}_{\Delta}$. We say that $\mathcal{A}$ *robustly satisfies* a property $\mathcal{P}$ if there exists positive values for the parameter Δ such that all the executions of $[\![\mathcal{A}]\!]^{\mathsf{AAsap}}_{\Delta}$ satisfy $\mathcal{P}$. It is shown in [DDR05] that the Almost-ASAP semantics $[\![\mathcal{A}]\!]^{\mathsf{AAsap}}_{\Delta}$ over-approximates the program semantics $[\![\mathcal{A}]\!]^{\mathsf{Prg}}_{\Delta_L,\Delta_P,\varepsilon}$ if $\Delta > 3\Delta_L + 4\Delta_P$ when $\varepsilon = 0$,[1] and therefore a timed automaton is implementable w.r.t. a property $\mathcal{P}$ if it robustly satisfies $\mathcal{P}$. In the rest of this paper, we denote by $[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}$ the semantics of $\mathcal{A}$ enlarged (or perturbed) by Δ and ε, *i.e.*, whose guards are enlarged by Δ and whose clocks may drift by at most ε.

The robust semantics considered here differs from the Almost-ASAP semantics. First, the Almost-ASAP semantics does not consider drifting clocks, but as we will see later, enlargements on guards only are sufficient in the sense that guard perturbations and drifts on clocks give rise to the same reachable states. Second, in the Almost-ASAP semantics, we make a distinction between the controller and the plant: only the controller is enlarged. Here we consider enlargements of all the transitions, even those that belong to the plant. So, clearly, the robust semantics that we consider here over-approximates the AASAP semantics, and robust correctness considered here implies implementability in the sense of [DDR05].

Robust verification of timed automata. In this paper, we consider the *robust safety verification problem* which asks, given a timed automaton $\mathcal{A}$ and a set of Bad states to avoid (*i.e.*, a *safety* property), if there exist $\Delta, \varepsilon \in \mathbb{Q}_{>0}$ such that $\mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}) \cap \mathsf{Bad} = \varnothing$. The main result of this paper is to show that the robust safety verification problem is decidable.[2] To show this, we make a strong link with the robust semantics defined in [Pur98, Pur00] where Puri presents an algorithm to compute the set $\cap_{\varepsilon>0}\mathsf{Reach}([\![A]\!]^{\varepsilon}_{0})$. This is the set of states that can be reached when the clocks drift by an infinitesimal amount. We show that Puri's algorithm can be used to compute the following sets (which are therefore equal):

$$\bigcap_{\varepsilon>0} \mathsf{Reach}([\![A]\!]^{\varepsilon}_{0}) = \bigcap_{\Delta>0} \mathsf{Reach}([\![A]\!]^{0}_{\Delta}) = \bigcap_{\Delta>0}\bigcap_{\varepsilon>0} \mathsf{Reach}([\![A]\!]^{\varepsilon}_{\Delta})$$

The proof of this result follows the general ideas of Puri's proof and it is based on the structure of limit cycles of timed automata (a fundamental notion introduced by Puri) but we need new techniques to handle both the imprecisions on guards and the clock drifts. To establish the decidability of the robust safety verification problem, we show that

$$\bigcap_{\Delta>0}\bigcap_{\varepsilon>0} \mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}) \cap \mathsf{Bad} = \varnothing \qquad \text{iff} \qquad \exists \Delta > 0,\ \varepsilon > 0.\ \mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}) \cap \mathsf{Bad} = \varnothing.$$

Hence, to solve the robust safety verification problem, it suffices to compute the set $\bigcap_{\Delta>0}\bigcap_{\varepsilon>0} \mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta})$ and check if it has an empty intersection with Bad. We give a detailed proof of all intermediate results, some of which are useful by themselves to develop the robust verification of timed automata for LTL and a fragment of MTL [BMR06, BMR08]. This paper is an extended and revised version of [DDMR04].

[1] For $\varepsilon > 0$, a general constraint of the form $\Delta > f(\Delta_L, \Delta_P, \varepsilon)$ is also established in [DDR05], where f is a simple function such that $f(\Delta_L, \Delta_P, \varepsilon) \to 0$ when $\Delta_L, \Delta_P, \varepsilon \to 0$.

[2] This holds under some assumption that is discussed in Section 2.

Related works. The problem of designing controllers for embedded systems is an extremely important subject and a very active area of research. This design problem has been addressed by a large number of researchers in computer science but also in other research communities like control theory. Researchers in control theory have studied the general problem of going from a continuous formulation of the control problem to a discrete solution which is implementable by digital devices. Here, we focus on the particular nice mathematical model of timed automata. Timed automata are important but by no means the only important mathematical model in the context of the model-based design of embedded control systems (see for instance Matlab-Simulink models, dynamical systems of control theory, etc.) Timed automata are adequate for modeling a large variety of real-time constrained behaviors but they abstract away other properties of systems, like power-consumption, distribution, concurrency etc., properties that may be important for the correct physical realization of timed controllers. Our paper thus focuses on an important aspect related to the implementation of real-time behaviors but other important aspects must be tackled by other techniques and are treated elsewhere (and should not be ignored in practice). In the rest of this paragraph, we concentrate on other works that are related to the specific issue of the robustness of timed models and their realization by digital devices.

One of the first attempts to make mathematical models of timed systems implementable was done by Dierks for Programmable Logic Controllers (PLC) [Die99, Die01]. The language used to define the semantics of PLC-automata and to specify real-time constraints is the Duration Calculus [CHR91], which is a dense time interval-based temporal logics that allows to fix the delays for the visibility of events, the computation times, the imprecision of the clocks, etc. That logic is very expressive, but its major drawback is to be undecidable [CHS93].

Other notions of robustness for timed systems have also been addressed in several works for timed automata [GHJ97, ALM05] and hybrid automata [Frä99, AB01, AT04], but none of these works make a link between robustness and implementability.

In [HKSP03], 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 as an intermediary step between mathematical models like hybrid automata and real execution platforms.

In [AIK+03], 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. In [AFM+02, AFP+03], Yi *et al.* present a tool called TIMES, which generates executable code from timed automaton 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 [AT05], Altisen and Tripakis tackle the problem of the implementability of timed automata specifications by considering an alternative direction to ours. By contrast with our approach, they do consider the usual semantics of timed automata and propose to model the execution platform within this formal semantics. For instance, they show how drifting clocks or delays in synchronization can be modeled using parametrized widgets in the form of timed automata. The resulting model is a network of timed automata composed of the initial model of the controller and a bunch of timed automata that models the platform. On the one hand, this approach is general in the sense that different implementation platforms can be modeled accurately by timed automata models. On the other hand, one drawback of their approach comes from the fact that models of specific platforms tend to be very large and may impair the automatic verification step because of the state explosion problem. Also, in early steps of the design of a timed controller, we may not know the exact platform that will be used to implement the control strategy but we may already be interested to verify the robustness of the control strategy with regard to timing imprecisions. The use of a robust semantics partially avoid those problems by leaving the details of the implementation unspecified: the proof of implementability makes reference to a naive implementation schema which imposes very few constraints on the platform that is chosen for the actual implementation and which is met by any reasonable implementation platform (see [DDR05] for more details). The price to pay in our setting is that we may be over-pessimistic and declare some control strategies not to be implementable while a precise modeling of the execution platform in the setting of Altisen and Tripakis might allow to establish implementability of the control strategy.

Finally, note that the algorithm presented in this paper to solve the robust safety verification problem is not usable in practice as it relies on the construction of the region automaton. Recent works have investigated the development of practical algorithms [DK06,SF07] but, to the best of our knowledge, no efficient implementation has been released yet.

2 Timed models

This section is devoted to the definitions of timed automata and their perturbed semantics, and of several other important notions that will be used in this paper.

In the sequel, $\mathbb{R}_{\geqslant 0}$ is the set of nonnegative reals, and $\mathbb{N}$ is the set of nonnegative integers.

Definition 1 *A* timed transition system *(TTS for short) $\mathcal{T}$ 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 $\rightarrow \subseteq S \times (\Sigma \cup \mathbb{R}_{\geqslant 0}) \times S$ is the transition relation. We write $q \xrightarrow{\sigma} q'$ if $(q, \sigma, q') \in \rightarrow$.*

A *trajectory* of a TTS $\mathcal{T} = \langle S, \iota, \Sigma, \rightarrow\rangle$ is a finite sequence $\pi = (s_0, t_0) \xrightarrow{\sigma_1} (s_1, t_1) \ldots \xrightarrow{\sigma_k} (s_k, t_k)$ such that for all $0 \leqslant i \leqslant k$, we have $(s_i, t_i) \in S \times \mathbb{R}_{\geqslant 0}$, and for all $0 \leqslant i < k$, we have $s_i \xrightarrow{\sigma_i} s_{i+1}$, and either $\sigma_i \in \Sigma$ and $t_{i+1} = t_i$, or $\sigma_i \in \mathbb{R}_{\geqslant 0}$ and $t_{i+1} = t_i + \sigma_i$.

Let $\pi = (s_0, t_0) \xrightarrow{\sigma_1} (s_1, t_1) \ldots \xrightarrow{\sigma_k} (s_k, t_k)$ be a trajectory. The $i+1$-st state in π, written $\mathsf{state}_i(\pi)$, is the state s_i. We denote by $\mathsf{first}(\pi) = \mathsf{state}_0(\pi)$ and $\mathsf{last}(\pi) = \mathsf{state}_k(\pi)$ the initial and final states of π. We say that π is a trajectory from s_0 to s_k, and we sometimes write π more briefly as $s_0 \xrightarrow{\sigma_1} s_1 \ldots \xrightarrow{\sigma_k} s_k$. The *length* of π, written $|\pi|$, is k, and its *duration* $\mathsf{Duration}(\pi)$ is $t_k - t_0$. The sequence of states $s_0\, s_1 \ldots s_k$ is called a *path* in $\mathcal{T}$, and the *trace* of π is the sequence $\mathsf{trace}(\pi) = \lambda_1 \ldots \lambda_k$ where for all $1 \leqslant i \leqslant k$, $\lambda_i = \sigma_i$ if $\sigma_i \in \Sigma$, and $\lambda_i = \mathsf{time}$ if $\sigma_i \in \mathbb{R}_{\geqslant 0}$ (time is a special symbol not in Σ).

A trajectory is *stutter-free* iff its trace contains alternately a symbol in Σ and the symbol time, *i.e.*, it does not contain two consecutive symbols in Σ or two consecutive time's.

A state s' of $\mathcal{T}$ is *reachable* from a state s if there exists a trajectory π of $\mathcal{T}$ such that $\mathsf{first}(\pi) = s$ and $\mathsf{last}(\pi) = s'$. Given a set of states $Q \subseteq S$, we write $\mathsf{Reach}(\mathcal{T}, Q)$ for the set of states that are reachable from some state in Q. We abusively write $\mathsf{Reach}(\mathcal{T})$ for $\mathsf{Reach}(\mathcal{T}, \{\iota\})$, the set of states that are reachable from the initial state of $\mathcal{T}$.

Given a set $\mathsf{Var} = \{x_1, \ldots, x_n\}$ of clocks, a *clock valuation* is a function $v\colon \mathsf{Var} \to \mathbb{R}_{\geqslant 0}$. In the sequel, we often identify a clock valuation with a point in $\mathbb{R}^n_{\geqslant 0}$. Given two valuations v and v' and two nonnegative reals t and λ, we write $v + v'$ for the valuation $x \mapsto v(x) + v'(x)$, $v + t$ for the valuation $x \mapsto v(x) + t$ and λv for $x \mapsto \lambda v(x)$. If $R \subseteq \mathsf{Var}$, then $v[R := 0]$ denotes the valuation v' such that $v'(x) = 0$ for all $x \in R$, and $v'(x) = v(x)$ for all $x \notin R$.

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

We now define timed automata. Our definition is a slightly modified version of the classical timed automata proposed by [AD94]. In particular, all clocks are assumed to be bounded by some constant M, and guards on edges are rectangular and closed. The first requirement is not restrictive (except w.r.t. the conciseness of the models [BC05]), the second one is discussed below.

Definition 2 *A* (closed) timed automaton *is a tuple* $\mathcal{A} = \langle \mathit{Loc}, \mathit{Var}, q_0, \mathit{Lab}, \mathit{Edg} \rangle$ *where*

- Loc *is a finite set of locations representing the discrete states of* $\mathcal{A}$.
- $\mathit{Var} = \{x_1, \ldots, x_n\}$ *is a finite set of real-valued variables.*
- $q_0 = (l_0, v_0)$, *where* $l_0 \in \mathit{Loc}$, *is the initial location and* v_0 *is the initial clock valuation such that for any* $x \in \mathit{Var}$, $v_0(x) \in \mathbb{N} \wedge v_0(x) \leqslant M$.
- Lab *is a finite alphabet of labels.*
- $\mathit{Edg} \subseteq \mathit{Loc} \times \mathit{Rect}_c(\mathit{Var}) \times \mathit{Lab} \times 2^{\mathit{Var}} \times \mathit{Loc}$ *is the set of transitions. A transition* (l, g, σ, R, l') *represents a jump from location* l *to location* l' *with guard* g, *event* σ *and a subset* $R \subseteq \mathit{Var}$ *of variables to be reset.*

We now define a family of semantics for timed automata which is parametrized by $\varepsilon \in \mathbb{R}_{\geqslant 0}$ (drift on clocks) and $\Delta \in \mathbb{R}_{\geqslant 0}$ (imprecision on guards).

Definition 3 *Given* $\varepsilon, \Delta \in \mathbb{R}_{\geqslant 0}$, *the* perturbed semantics *of a timed automaton* $\mathcal{A} = \langle \mathit{Loc}, \mathit{Var}, q_0, \mathit{Lab}, \mathit{Edg} \rangle$ *is the TTS* $[\![\mathcal{A}]\!]^\varepsilon_\Delta = \langle S, \iota, \Sigma, \to \rangle$ *where:*

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

The traditional semantics of $\mathcal{A}$ is $[\![\mathcal{A}]\!]^0_0$. When $\varepsilon > 0$ or $\Delta > 0$, we call $[\![\mathcal{A}]\!]^\varepsilon_\Delta$ the perturbed semantics of $\mathcal{A}$. In the sequel, we often write $[\![\mathcal{A}]\!]$, $[\![\mathcal{A}]\!]_\Delta$, and $[\![\mathcal{A}]\!]^\varepsilon$ instead of respectively $[\![\mathcal{A}]\!]^0_0$, $[\![\mathcal{A}]\!]^0_\Delta$, and $[\![\mathcal{A}]\!]^\varepsilon_0$.

Remark 1 Our definition of timed automata does not allow strict inequalities; This is not restrictive in the presence of guard enlargement. Indeed, consider a timed automaton $\mathcal{A}$ with (possibly open) rectangular guards and the closure automaton $\widehat{\mathcal{A}}$ resulting from $\mathcal{A}$ by replacing all strict inequalities by non-strict ones. It appears obviously that

$$\mathsf{Reach}([\![\widehat{\mathcal{A}}]\!]^{\varepsilon}_{\frac{\Delta}{2}}) \subseteq \mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}) \qquad \text{and} \qquad \mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}) \subseteq \mathsf{Reach}([\![\widehat{\mathcal{A}}]\!]^{\varepsilon}_{\Delta}),$$

and hence the robust safety verification problem for $\mathcal{A}$ and Bad ("*Do there exist* $\Delta, \varepsilon \in \mathbb{R}_{>0}$ *such that* $\mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}) \cap \mathsf{Bad} = \varnothing$*?*") is equivalent to the the robust safety verification problem for $\widehat{\mathcal{A}}$ and Bad. Note that this only holds thanks to guard enlargement, and the situation is different when only clock drifts are allowed [Dim07].

Remark 2 Notice that guard enlargement and clock drifts are *monotone*, in the sense that for any $\Delta \leqslant \Delta'$ and $\varepsilon \leqslant \varepsilon'$, we have $\mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}) \subseteq \mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon'}_{\Delta'})$. Thanks to this observation, we can define the following sets:

$$R^{\varepsilon}_{\Delta\to 0} = \bigcap_{\Delta>0} \mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}) \qquad\qquad R^{\varepsilon\to 0}_{\Delta} = \bigcap_{\varepsilon>0} \mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta})$$

These sets are central in the sequel: we prove that, under some minor restriction, the sets $R^{0}_{\Delta\to 0}$ and $R^{\varepsilon\to 0}_{0}$ are equal, and they also coincide with $\bigcap_{\Delta>0} R^{\varepsilon\to 0}_{\Delta}$ and $\bigcap_{\varepsilon>0} R^{\varepsilon}_{\Delta\to 0}$. It follows from [Pur98] that those sets are computable; we will reprove this result in the sequel.

We now recall some additional classical notions related to timed automata. In the sequel, $\lfloor x \rfloor$ denotes the integer part of x (the greatest integer $k \leqslant x$), and $\langle x \rangle$ denotes its fractional part.

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

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

We denote by (v) the clock region containing v, and by $[r]$ the topological closure of a region r, which is then abusively called a *closed clock region.* The clock region (v) contains the valuations that agree with v on the integer part of the variables, and on the ordering of their fractional part and zero.

A clock region r' is a *time-successor* of a clock region r if $r' \neq r$ and for some $v \in r$ and $t \in \mathbb{R}_{>0}$, we have $v + t \in r'$.

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

- $R_{\mathcal{A}} = \{(l, r) \mid \exists (l, v) \in S : r = (v)\}$ *is the set of* regions*;*
- $\to_G \subseteq R_{\mathcal{A}} \times (\Sigma \cup \{\mathsf{time}\}) \times R_{\mathcal{A}}$ *where* $((l, r), \mathsf{time}, (l', r)) \in \to_G$ *if and only if* $l' = l$ *and r' is a time-successor of r, and* $((l, r), \sigma, (l', r)) \in \to_G$ *for* $\sigma \in \Sigma$ *if and only if* $(l, v) \xrightarrow{\sigma}_{\mathcal{A}} (l', v')$ *for some $v \in r$ and $v' \in r'$.*

Notice that $R_{\mathcal{A}}$ is finite (the total number $W = |R_{\mathcal{A}}|$ of regions is exponential in the size of $\mathcal{A}$) and a region (l', r') is reachable in G from a region (l, r) if and only if (l', v') is reachable in $[\![\mathcal{A}]\!]$ from a state (l, v) for some $v \in r, v' \in r'$ [AD94].

In the rest of the paper, we often abusively use operators that apply to valuations v or clock regions r to pairs (ℓ, v) and (ℓ, r) where ℓ is a location. For example, we write $(\ell, v) + (\ell, v')$ to denote $(\ell, v + v')$ and $\lambda(\ell, v)$ to denote $(\ell, \lambda v)$, and similarly for distances, norms and neighbourhoods. We write $(\ell, v) \in (\ell, r)$ instead of $v \in r$, $[(\ell, v)]$ instead of $(\ell, [v])$, etc.

Since guards are closed, the successors of a closed region by a discrete transition or by the passage of time is a union of closed regions. Since there are finitely many regions, the next lemma follows.

Lemma 6 *Let $\mathcal{A}$ be a (closed) timed automaton. The set $\mathsf{Reach}([\![\mathcal{A}]\!])$ is a closed set.*

Given a path $p = p_0 p_1 \cdots p_N$ in the region graph of a timed automaton $\mathcal{A}$, and a trajectory π of $[\![\mathcal{A}]\!]$, we say that π *follows* p if for all i, $0 \leqslant i \leqslant N$, $\mathsf{state}_i(\pi) \in [p_i]$. Note that, since we consider closed regions, a trajectory could follow several paths of the region graph.

Definition 7 *A* zone *$Z \subseteq \mathbb{R}^n_{\geqslant 0}$ is a closed set defined by inequalities of the form*

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

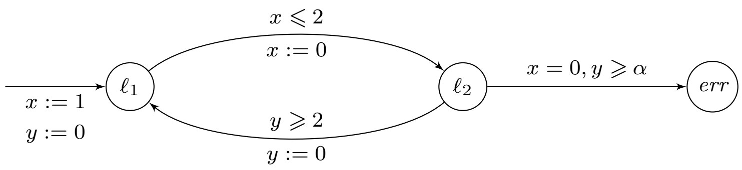
where $1 \leqslant i, j \leqslant n$ and $m_{ij}, \alpha_i, \beta_i \in \mathbb{Z}$. A set of states is called a zone-set *if it is a finite union of sets of the form $\{l\} \times Z$ where l is a location and Z is a zone.*

Definition 8 *A* progress cycle *in the region graph of a timed automaton is a cycle in which each clock of the automaton is reset at least once.*

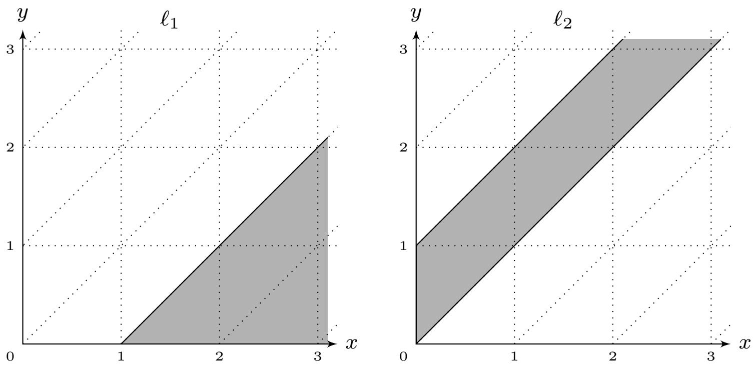
The correctness of our algorithm (Theorem 11 below) heavily relies on the fact that timed automata should not have *weird* behaviors, like cycles that do not let time elapse. However, apart from Theorems 11, 39 and 47, all our intermediate results hold in the general case. We thus formulate the following assumption, but we will always explicitly refer to it when it is needed.

Assumption 9 *We only consider timed automata whose cycles in the region graph are all progress cycles.*

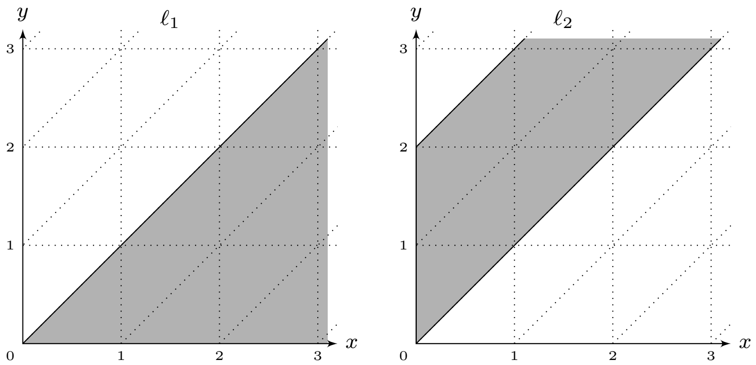
This assumption was made by Puri in [Pur98]. It is weaker than the classical non-Zeno assumptions in the literature. For example in [AMPS98], the authors impose that "in every cycle in the transition graph of the automaton, there is at least one transition which resets a clock variable x_i to zero, and at least one transition which can be taken only if $x_i \geqslant 1$". Other natural hypotheses would be to ask that every cycle in the region graph has a timed transition (labeled by time), or that every cycle in the region graph contains a *time-elapsing* region, that is, a region r such that $\exists v \in r \cdot \exists t > 0 : v + t \in r$. Again, Assumption 9 is weaker.



(a) A timed automaton $\mathcal{A}_\alpha$.



(b) $\mathsf{Reach}([\![\mathcal{A}_\alpha]\!])$ for timed automaton $\mathcal{A}_\alpha$.



(c) The set $\bigcap_{\Delta>0} \mathsf{Reach}([\![\mathcal{A}_\alpha]\!]^0_\Delta)$ for timed automaton $\mathcal{A}_\alpha$.

Fig. 1 Differences between standard and perturbed semantics.

Example Consider the timed automaton $\mathcal{A}_\alpha$ of Fig. 1(a) where $\alpha \in \{2, 3\}$. The automaton has two clocks x and y. There is one initial location ℓ_1 with initial clock values $x = 1$ and $y = 0$. For locations ℓ_1 and ℓ_2, the sets of reachable states in the classical semantics $[\![\mathcal{A}_\alpha]\!]$ with $\varepsilon = \Delta = 0$ are depicted in Fig. 1(b). The final states (or bad states) correspond to the location *err* with any clock valuation. For both $\alpha = 2$ and $\alpha = 3$, the timed automaton $\mathcal{A}_\alpha$ does not reach the bad states.

Consider the perturbed semantics $[\![\mathcal{A}_\alpha]\!]^0_\Delta$ for $\varepsilon = 0$ and $\Delta > 0$. In this semantics, guards are enlarged by Δ. The edge from ℓ_1 to ℓ_2 has the guard $x \leqslant 2 + \Delta$ and the edge from ℓ_2 to ℓ_1 has the guard $y \geqslant 2 - \Delta$. From the initial state $(\ell_1, x = 1, y = 0)$, the transition to ℓ_2 can be taken after Δ time units, reaching the states $(\ell_2, x = 0, y \leqslant 1 + \Delta)$. Similarly, the transition from ℓ_2 back to ℓ_1 is enabled Δ time units earlier than before and the states $(\ell_1, x \geqslant 1-2\Delta, y=0)$ are reachable. It is easy to show that after having taken k times the transitions of the cycle, the states $(\ell_1, x \geqslant 1-2k\Delta, y=0)$ (provided $x \geqslant 0$) and $(\ell_2, x=0, y \leqslant 1+(2k-1)\Delta)$ (provided $y \leqslant 2$) are reachable. Hence, for all $\Delta > 0$ the states $(\ell_1, x \geqslant 0, y=0)$ and $(\ell_2, x=0, y \leqslant 2)$ are reachable in $[\![\mathcal{A}_\alpha]\!]^0_\Delta$ and were not reachable in the classical semantics $[\![\mathcal{A}_2]\!]^0_0$. Those states are represented in Fig. 1(c). The same situation occurs in the perturbed semantics $[\![\mathcal{A}_\alpha]\!]^\varepsilon_0$ for $\Delta = 0$ and $\varepsilon > 0$, that is, $R^{\varepsilon\to 0}_\Delta = R^\varepsilon_{\Delta\to 0} \neq \mathsf{Reach}([\![\mathcal{A}_\alpha]\!])$.

This example shows that the classical semantics is not robust with respect to small perturbations in either the timing constraints or the clock rate. The effect of such perturbations, no matter how small they are, may lead to dramatically different behaviours of the system. In this case, the location *err* is reachable in the perturbed semantics $[\![\mathcal{A}_2]\!]^0_\Delta$ for any $\Delta > 0$, but not in the classical semantics $[\![\mathcal{A}_2]\!]^0_0$. We say that the safety property (to avoid the location *err*) is not robustly satisfied by $\mathcal{A}_2$. Such non-robust systems cannot have a correct implementation because their correctness relies on the mathematical idealization of the traditional semantics.

On the other hand, for $\alpha = 3$ the safety property still holds in the limit of the perturbed semantics. As we will show, this implies that there exists a strictly positive value of Δ for which the perturbed semantics $[\![\mathcal{A}_\alpha]\!]^0_\Delta$ is safe. In fact, any $\Delta < \frac{1}{3}$ fits in our example.

3 The robust safety verification problem is decidable

The main result of this paper is a detailed proof that the robust safety verification problem is decidable, for both perturbed guards and drifting clocks (Theorem 11).

Definition 10 *The* robust safety verification problem *asks, given a timed automaton $\mathcal{A}$ and a zone-set Bad, if there exist $\Delta, \varepsilon \in \mathbb{Q}_{>0}$ such that $\mathsf{Reach}([\![\mathcal{A}]\!]^\varepsilon_\Delta) \cap \mathit{Bad} = \varnothing$.*

Theorem 11 *Under assumption 9, the robust safety verification problem is decidable.*

The rest of the paper is devoted to the detailed proof of this result. The main argument is based on an algorithm presented by Puri in [Pur98, Pur00] to compute a robust semantics for timed automata, namely the set $R^{\varepsilon\to 0}_0$. Puri gives a proof of correctness of his algorithm, using innovative techniques for the analysis of cycles in timed automata. However, some of Puri's important results are proven in broad outline. For instance, Puri's claim that $R^{\varepsilon\to 0}_0 = R^{\varepsilon\to 0}_{\Delta\to 0}$ where $R^{\varepsilon\to 0}_{\Delta\to 0} = \bigcap_{\Delta>0}\bigcap_{\varepsilon>0} \mathsf{Reach}([\![\mathcal{A}]\!]^\varepsilon_\Delta)$ is not really justified ([Pur98, Theorem 9.1]). Moreover, several proofs of intermediate results that are used by Puri to establish the correctness of his algorithm are quickly sketched, giving a rough idea of the proof scheme. For some of them, we were not able to complete the proofs, in particular for [Pur98, Lemma 6.4] where the statement of the lemma itself is actually wrong.

Therefore, we give in the next sections a detailed proof of Theorem 11 and of the equality $R^{\varepsilon\to 0}_{\Delta\to 0} = R^{\varepsilon\to 0}_0$. Moreover, we show that the same semantics is also obtained under guard perturbations only, *i.e.*, $R^0_{\Delta\to 0} = R^{\varepsilon\to 0}_{\Delta\to 0} = R^{\varepsilon\to 0}_0$. On the one hand, we exploit the new techniques introduced by Puri and we clarify and reprove some of his lemmas, and on the other hand, we also introduce new lemmas and proof techniques to bridge over some gaps of Puri's papers, and extend his results to the semantics with guard perturbations. We proceed with the following steps:

1. in Section 4, we recall classical results about regions and zones for timed automata. We essentially follow the work of Puri;
2. in Section 5, we show that the robust safety verification problem reduces to the computation of the set $R_{\Delta\to 0}^{\varepsilon\to 0} = \bigcap_{\Delta>0}\bigcap_{\varepsilon>0}\mathsf{Reach}([\![\mathcal{A}]\!]_{\Delta}^{\varepsilon})$ (Theorem 15). This has no counterpart in [Pur98].
3. in Section 6, we show that Algorithm 1 below (due to Puri) computes $R_{\Delta\to 0}^{\varepsilon\to 0} = R_{\Delta\to 0}^{0} = R_{0}^{\varepsilon\to 0}$:
   (a) in Section 6.1, we study the structure of the cycles of timed automata and of their region graph. This is important because cycles allow perturbations to accumulate (see the example of Section 2). The material in this section is essentially due to Puri, and the detailed proofs follow his ideas;
   (b) in Section 6.2, we show that Algorithm 1 is sound, *i.e.*, the set J^* it computes is contained in $R_{\Delta\to 0}^{0}$ and in $R_{0}^{\varepsilon\to 0}$ (Theorem 39). This part required 3 pages for $R_{\Delta\to 0}^{0}$ and 5 pages for $R_{0}^{\varepsilon\to 0}$, to give a precise justification of the 5 lines in [Pur00, Lemma 7.11];
   (c) in Section 6.3, we show that Algorithm 1 is complete, *i.e.*, the set J^* it computes contains $R_{\Delta\to 0}^{\varepsilon\to 0}$ (Theorem 46). Our approach to establish the key lemmas of [Pur00, Section 8.2] uses an original technique based on parametric DBMs, which allows to extend Puri's results to both guard perturbations and clock drifts;
4. in Section 7, we show that the safety verification problem is PSPACE-complete, as claimed by Puri.

Several results in this paper are highly technical, and all proofs are given in details. To help the reader, we give overviews of the main technical developments in the beginning of Sections 6, 6.2 and 6.3.

4 Properties of Regions and Zones

We review the important properties of the clock regions and zones of timed automata, for a heavy use in the sequel.

Clock regions. According to Definition 4, a clock region of a timed automaton contains a set of valuations that agree on the integral part of the clocks and on the ordering of their fractional parts. We make this characterization of clock regions more concrete with the following representation [Pur98].

Definition 12 *Given a timed automaton $\mathcal{A}$ with n clocks ($\mathsf{Var} = \{x_1, \ldots, x_n\}$) and largest constant M, we represent a clock region of $\mathcal{A}$ by:*

1. *a tuple of $(a_1, \ldots, a_n)$ of elements of $\{0, 1, ..., M, \perp\}$;*
2. *and a tuple $(X_0, X_1, \ldots, X_k)$ of $k+1$ ($0 \leqslant k \leqslant n$) sets of clocks that form a partition of the clocks that have a value less than M. Those sets are required to be non-empty, except X_0. Formally, let $\mathsf{Var}^{\leqslant M} = \{x_i \in \mathsf{Var} \mid a_i \neq \perp\}$. We require that $\mathsf{Var}^{\leqslant M} = X_0 \cup \cdots \cup X_k$, $X_i \cap X_j = \varnothing$ if $i \neq j$ and $X_i \neq \varnothing$ for all $1 \leqslant i \leqslant k$.*

The clock region characterized by a tuple $(a_i)_{1\leqslant i\leqslant n}$ and $(X_i)_{0\leqslant i\leqslant k}$ is the set of all valuations $v : \mathsf{Var} \to \mathbb{R}_{\geqslant 0}$ such that:

1. *For all $x_i \in \mathsf{Var}$: $v(x_i) > M$ iff $a_i = \perp$, and if $a_i \neq \perp$ then $\lfloor v(x_i)\rfloor = a_i$;*

2. for all $x, y \in \mathsf{Var}^{\leqslant M}$: $\langle v(x)\rangle < \langle v(y)\rangle$ iff for some $i < j$, $x \in X_i$ and $y \in X_j$;
3. and for all $x \in \mathsf{Var}^{\leqslant M}$: $\langle v(x)\rangle = 0$ iff $x \in X_0$.

A more classical way to represent clock regions is by the set of constraints it satisfies. Our representation of a clock region r is easily translated to a set of constraints that are satisfied by (and only by) valuations of r. A valuation v belongs to the clock region represented by $(a_i)_{1\leqslant i\leqslant n}$ and $(X_i)_{0\leqslant i\leqslant k}$ if and only if v satisfies the following constraints [AD94]:

- $x_i > M$ for each x_i such that $a_i = \perp$;
- $x_i = a_i$ for each $x_i \in X_0$;
- $a_i < x_i < a_i + 1$ for each $x_i \in X_l$ for some $l > 0$;
- $x_i - a_i < x_j - a_j$ for each x_i, x_j such that $x_i \in X_l$ and $x_j \in X_m$ for some $0 < l < m$;
- $x_i - a_i = x_j - a_j$ for each $x_i, x_j \in X_l$ for some $l > 0$.

Example. In a timed automaton with 5 clocks and largest constant $M = 8$, a clock region r represented by $(1, 3, 5, \perp, 2)$ and $(\{x_1, x_3\}, \{x_2\}, \{x_5\})$ satisfies the following constraints:

$$0 = x_1 - 1 = x_3 - 5 < x_2 - 3 < x_5 - 2 < 1 \wedge x_4 > 8.$$

The closure $[r]$ of r then satisfies:

$$0 = x_1 - 1 = x_3 - 5 \leqslant x_2 - 3 \leqslant x_5 - 2 \leqslant 1 \wedge x_4 \geqslant 8.$$

Vertices of a clock region. For a set $S \subseteq \mathbb{R}^n$, let $\mathsf{Conv}(S)$ be the convex hull of S, *i.e.*, the smallest convex set containing S. Since clock regions are bounded, they are convex polytopes, and can also be defined as the convex hull of a finite set of points:

Definition 13 *Let r be a clock region of a timed automaton. The set of* vertices *of r is the smallest set of points $S(r)$ such that $[r] = \mathsf{Conv}(S(r))$.*

Lemma 14 shows that the set $S(r)$ is unique and the number of vertices is at most $n + 1$, where n is the number of clocks of the automaton.

Lemma 14 *The vertices of a clock region r are the integer vectors of its closure: $S(r) = [r] \cap \mathbb{N}^n$.*

Proof. Let $v \in [r]$. Let the representation of r be given as $(a_i)_{1\leqslant i\leqslant n}$ and $(X_i)_{0\leqslant i\leqslant k}$. Then, for all valuations $w \in [r]$ we have:

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

Let v_0 be the valuation such that $v_0(x_i) = a_i$ for each $0 \leqslant i \leqslant k$. We have $v_0 \in [r]$ and for all $0 < j \leqslant k$, the valuation v_j defined by

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

belongs to $[r]$.

We now prove that those valuations generate the whole closed clock region $[r]$. Let w be a valuation in $[r]$. We define

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

where each x_i is a clock in the corresponding X_i for $1 \leqslant i \leqslant k$, and $w(x_0) = 0$ by convention. We claim that $w' = w$. To show this, let y be a clock in some X_j. Then

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

since $y \in X_j$. Therefore $[r] = \mathsf{Conv}(\{v_0, \ldots, v_k\})$. On the other hand, $\{v_0, \ldots, v_k\}$ is the smallest set generating $[r]$, since there is no valuation v_i that is a convex combination of the others. ■

Lemma 14 entails that if a clock region r is a sub-region of a clock region r', then its set of vertices $S(r)$ is the intersection of $[r]$ and the set $S(r')$ of vertices of r'.

Zones and DBMs. Related to the algorithmic analysis of timed automata, an efficient data structure has been introduced to represent zones: the difference bound matrices (DBM) [BM83, Dil90]. We briefly introduce DBMs and show how the basic operations that are useful for reachability analysis are computed[3].

Let $x_1, \ldots, x_n$ be the clocks of a timed automaton. The idea of DBMs is to represent all constraints uniformly, by constraints of the form $x_i - x_j \leqslant a$ with $a \in \mathbb{Z} \cup \{+\infty\}$ and $0 \leqslant i, j \leqslant n$ where x_0 is the constant 0. For bounded zones, the range of a can be reduced to $\mathbb{Z} \cap [-M, M]$ where M is the largest constant of the timed automaton.

A DBM is a $(n+1) \times (n+1)$ matrix $\mathbf{M} = \big(m_{i,j}\big)_{0 \leqslant i,j \leqslant n}$ where each $m_{i,j}$ is of the form $(a_{i,j}, \prec_{i,j})$ where $\prec_{i,j} \in \{<, \leqslant\}$ and $a_{i,j} \in \mathbb{Z}$ is called a *bound*. In the sequel, we only consider DBMs that represent closed sets, that is, DBMs where $\prec_{i,j}$ is always $\leqslant$. The set of valuations represented by the DBM $\mathbf{M} = \big(m_{i,j}\big)_{0 \leqslant i,j \leqslant n}$ is:

$$[\![\mathbf{M}]\!] = \{(x_1, \ldots, x_n) \in \mathbb{R}^n \mid \forall\, 0 \leqslant i, j \leqslant n : x_i - x_j \leqslant m_{ij} \wedge x_0 = 0\}.$$

A DBM is associated with a complete directed graph with nodes $0, 1, \ldots, n$ and edges (i, j) labeled by m_{ij}. In this graph, the *length* of a path is the sum of the labels of the edges in the path. It is easy to see that the length of a path from node i to node j is an upper bound of the difference $x_i - x_j$. The length of the shortest paths between nodes gives the tightest bounds on the variables and difference of variables. This allows to define a *normal form* for DBMs that corresponds to the shortest path closure of the associated directed graph. If the graph contains a cycle of negative length, the shortest

[3] In Section 6.3, we study in more details a parametric extension of DBMs, which we use for proving completeness of our algorithm.

path closure does not exist and the DBM represents the empty set as a constraint of the form $x_i - x_i \leqslant m$ with $m < 0$ is unsatisfiable. Hence, for nonempty DBMs in normal form, we have $m_{i,i} = 0$ for all $0 \leqslant i \leqslant n$.

Operations on sets represented by DBMs are executed by syntactic transformations on the DBM. Some of those operations require the normal form. We present the operations that will be useful in the sequel. Other operations like the difference of two DBMs and the inclusion test are definable (see *e.g.* [ACD+92, Yov96, CGP99] for details).

- Intersection: given two DBMs $\mathbf{M} = (m_{i,j})_{0\leqslant i,j\leqslant n}$ and $\mathbf{M}' = (m'_{i,j})_{0\leqslant i,j\leqslant n}$, let $\mathbf{M}''$ be the DBM such that $m''_{i,j} = \min\{m_{i,j}, m'_{i,j}\}$. Then, we have $[\![\mathbf{M}'']\!] = [\![\mathbf{M}]\!] \cap [\![\mathbf{M}']\!]$. It is not required that $\mathbf{M}$ and $\mathbf{M}'$ are in normal form. In any case the result is not necessarily in normal form.
- Time passing: given a DBM in normal form $\mathbf{M} = (m_{i,j})_{0\leqslant i,j\leqslant n}$, let $\mathbf{M}\nearrow$ be the DBM $(m'_{i,j})_{0\leqslant i,j\leqslant n}$ such that $m'_{i,0} = \infty$ and $m'_{i,j} = m_{i,j}$ for all $0 \leqslant i,j \leqslant n$ with $j \neq 0$. This removes the upper bound on all the clocks. We have $[\![\mathbf{M}\nearrow]\!] = \{v + t \mid v \in [\![M]\!] \wedge t \in \mathbb{R}_{\geqslant 0}\}$ and $\mathbf{M}\nearrow$ is in normal form.
- Reset: given a DBM in normal form $\mathbf{M} = (m_{i,j})_{0\leqslant i,j\leqslant n}$ and a clock x, let $\mathbf{M}[x := 0]$ be the DBM $(m'_{i,j})_{0\leqslant i,j\leqslant n}$ such that for all $0 \leqslant i,j \leqslant n$ with $i \neq j$:

$$m'_{i,j} = \begin{cases} m_{0,j} & \text{if } x = x_i \\ m_{i,0} & \text{if } x = x_j \\ m_{i,j} & \text{otherwise} \end{cases}$$

  We have removed all the bounds involving x and set x to zero. We have $[\![\mathbf{M}[x := 0]]\!] = \{v[x := 0] \mid v \in [\![M]\!]\}$. We define similarly $\mathbf{M}[R := 0]$ for $R \subseteq \{x_1, \ldots, x_n\}$. The result is in normal form.
- Emptiness test: given a DBM $\mathbf{M} = (m_{i,j})_{0\leqslant i,j\leqslant n}$, we have $[\![\mathbf{M}]\!] = \varnothing$ if and only if there is a cycle in the directed graph associated to $\mathbf{M}$ whose length is negative. The emptiness test is realized by the shortest path algorithm used to put DBMs in normal form.

5 Removing existential quantification

This first part of the proof of Theorem 11 consists in removing the existential quantification on Δ and ε. Given a timed automaton $\mathcal{A}$, we let $R^{\varepsilon\to 0}_{\Delta\to 0} = \bigcap_{\Delta>0}\bigcap_{\varepsilon>0} \mathsf{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta})$. We then have the following result:

Theorem 15 *For any timed automaton $\mathcal{A}$, any zone-set Bad, the following equivalence holds:*

$$R^{\varepsilon\to 0}_{\Delta\to 0} \cap \mathit{Bad} = \varnothing \qquad \textit{iff} \qquad \exists \Delta > 0,\ \varepsilon > 0 : \mathit{Reach}([\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}) \cap \mathit{Bad} = \varnothing.$$

The proof of Theorem 15 is based on several intermediate lemmas. Lemma 16 corrects a wrong claim of Puri about a lower bound on the distance between two zones with empty intersection. This bound is claimed to be $\frac{1}{2}$ in [Pur98, Lemma 6.4]. We show that $\frac{1}{n}$ is the tightest bound, where n is the number of clocks.

We introduce the following classical distances over $\mathbb{R}^n$:

$$d_\infty(x,y) = \|x - y\|_\infty = \max_{1\leqslant i\leqslant n}(|x_i - y_i|) \quad d_1(x,y) = \|x - y\|_1 = \sum_{1\leqslant i\leqslant n}(|x_i - y_i|).$$

Lemma 16 *Let $Z_1, Z_2 \subseteq \mathbb{R}^n$ be two zones such that $Z_1 \cap Z_2 = \varnothing$. For all $x \in Z_1$ and $y \in Z_2$, we have $d_\infty(x, y) \geqslant \frac{1}{n}$. This bound is tight.*

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

We consider two zones given by two DBMs in normal form: $Z_1 \equiv [\![(m_{i,j})]\!]$ and $Z_2 \equiv [\![(m'_{i,j})]\!]$. Since $Z_1 \cap Z_2 = \varnothing$, there must exist a "negative cycle":

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

where each term $m^{(\prime)}_{i,j}$ of the sum can be taken either in the matrix of Z_1 or in the matrix of Z_2. We may assume that at least one $m^{(\prime)}_{i,j}$ comes from Z_1 and one from Z_2 since otherwise Z_1 (or Z_2) would be empty and the result would hold vacuously.

Since for DBMs in normal form, we have $m_{a,b} + m_{b,c} \geqslant m_{a,c}$ for all indices a, b, c, we can merge any two *consecutive* $m_{i,j}$ into one while keeping the inequality. The same holds for $m'_{i',j'}$, and we can thus assume 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

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

Terms can be rearranged in this sum, yielding

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

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

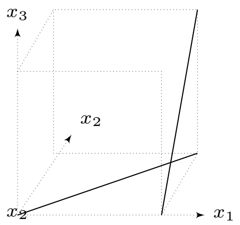
$$\begin{aligned}
1 &\leqslant \left|(y_{i_1} - x_{i_1}) - (y_{i_2} - x_{i_2}) + (y_{i_3} - x_{i_3}) - \cdots - (y_{i_p} - x_{i_p})\right| \\
&\leqslant |(y_{i_1} - x_{i_1})| + |(y_{i_2} - x_{i_2})| + |(y_{i_3} - x_{i_3})| + \cdots + |(y_{i_p} - x_{i_p})| \\
&\leqslant \|x - y\|_1.
\end{aligned}$$

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

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

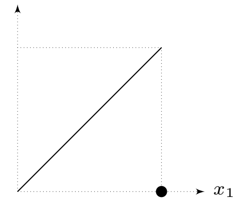
$$Z_2 \equiv \begin{cases} x_{2i-1} - x_{2i} = 0 & 1 \leqslant i \leqslant \frac{n-1}{2} \\ x_n = 0 & \end{cases}$$



– If n is even

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

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



We have $Z_1 \cap Z_2 = \varnothing$; indeed combining the equations of Z_1 and Z_2 yields $x_i = x_j$ for all $0 \leqslant i, j \leqslant n$, 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 $d_\infty(p, q) = \max(\frac{1}{n}, \dots, \frac{1}{n}) = \frac{1}{n}$. ■

In order to prove Lemma 16 in presence of both kinds of perturbation, we have to extend the previous lemma to sequences of sets as follows:

Lemma 17 *Let A_δ be a collection of sets such that $A_{\delta_1} \subseteq A_{\delta_2}$ if $\delta_1 \leqslant \delta_2$. Assume that $Z = \bigcap_{\delta > 0} A_\delta$ is a nonempty zone-set. Also assume the existence of a zone-set Z' such that $\exists \delta_0 > 0 \cdot \forall \delta \in (0, \delta_0) \cdot A_\delta \cap Z' = \varnothing$. Then there exists $\delta_1 > 0$ such that for all $0 < \delta < \delta_1$, we have $d_\infty(A_\delta, Z') \geqslant \frac{1}{2n}$.*

Proof. We pick $\delta_0 > 0$ such that $\forall 0 < \delta < \delta_0 \cdot\ A_\delta \cap Z' = \varnothing$, and $\delta_0' > 0$ such that

$$\forall x \in A_{\delta_0'} \cdot \exists z \in Z : d_\infty(x, z) < \frac{1}{2n}. \tag{1}$$

Such a δ_0' exists by definition of Z. Assume the lemma is wrong:

$$\forall \delta_1 > 0 \cdot \exists 0 < \delta < \delta_1 \cdot \exists x \in A_\delta,\ y \in Z' : d_\infty(x, y) < \frac{1}{2n}.$$

Applying this result with $\delta_1 = \min(\delta_0, \delta_0')$, we pick a $\delta_1' > 0$, and two points $x \in A_{\delta_1'}$ and $y \in Z'$ such that $d_\infty(x, y) < \frac{1}{2n}$. From (1), and since $A_{\delta_1'} \subseteq A_{\delta_0'}$, there exists $z \in Z$ such that $d_\infty(x, z) < \frac{1}{2n}$. Thus $d_\infty(y, z) < \frac{1}{n}$, and with Lemma 16, $Z \cap Z' \neq \varnothing$. Then any A_δ intersects Z', since it contains Z. This contradicts our hypotheses. ■

In the sequel, when a distance d or a norm $\|\cdot\|$ is used, we always refer to d_∞ and $\|\cdot\|_\infty$.

The following two lemmas rely on the theory of real numbers and the basics of topology.

Lemma 18 *If $d(A, B) > 0$, then $A \cap B = \varnothing$.*

Lemma 19 *If $A \subseteq B$, then $d(A, C) \geqslant d(B, C)$ for all C.*

Our main tool for proving Theorem 15 can then be stated as follows:

Lemma 20 *Let $A_\Delta (\Delta \in \mathbb{R}_{>0})$ be a collection of closed sets such that $A_{\Delta_1} \subseteq A_{\Delta_2}$ if $\Delta_1 \leqslant \Delta_2$. Assume that $A = \bigcap_{\Delta > 0} A_\Delta$ is nonempty. Let B be a bounded set. If $d(A, B) > 0$, then there exists $\Delta > 0$ such that $A_\Delta \cap B = \varnothing$.*

Define $\mathcal{N}_\infty(X, \eta) = \{x \in \mathbb{R}^n \mid \exists x' \in X : d_\infty(x, x') \leqslant \eta\}$ and let $\mathcal{N}_\infty(x, \eta) = \mathcal{N}_\infty(\{x\}, \eta)$.

Proof. For a contradiction, assume that for all $\Delta > 0$, we have $A_\Delta \cap B \neq \varnothing$. Let $\delta_i = \frac{1}{i}$ (for each $i \geqslant 1$). Then, we have:

$$\forall i \geqslant 1 \cdot \exists x_i \in A_{\delta_i} \cap B.$$

Since B is bounded, so is the set $\{x_i \mid i \geqslant 1\}$. By Bolzano-Weierstrass Theorem, there exists a point x such that:

$$\forall \varepsilon > 0 \cdot \forall i \geqslant 1 \cdot \exists j \geqslant i : x_j \in \mathcal{N}_\infty(x, \varepsilon) \cap A_{\delta_j}.$$

Let us show that x is in the closure of A_{δ_i} for all $i \geqslant 1$. Since $A_{\delta_j} \subseteq A_{\delta_i}$ for all $j \geqslant i$, we have:

$$\forall i \geqslant 1 \cdot \forall \varepsilon > 0 \cdot \exists j : x_j \in \mathcal{N}_\infty(x, \varepsilon) \cap A_{\delta_i},$$

that is:

$$\forall i \geqslant 1 \cdot \forall \varepsilon > 0 : \mathcal{N}_\infty(x, \varepsilon) \cap A_{\delta_i} \neq \varnothing.$$

Hence, for all $i \geqslant 1$ the point x is in the closure of the closed set A_{δ_i}, and thus $x \in A_{\delta_i}$. Since, for all $\Delta > 0$ there exists $i \geqslant 1$ such that $\delta_i \leqslant \Delta$ and thus $A_{\delta_i} \subseteq A_\Delta$, we have $\forall \Delta \in \mathbb{R}_{>0} : x \in A_\Delta$. This entails that $x \in A$. Now observe that:

$$\forall i \geqslant 1 : d_\infty(\{x\}, B) \leqslant d_\infty(x, x_i) + d_\infty(\{x_i\}, B).$$

Observe that $d_\infty(x, x_i)$ can be made arbitrarily small for sufficiently large i, and that for all $i \geqslant 1$, $d_\infty(\{x_i\}, B) = 0$ since $x_i \in B$. Therefore, we get:

$$\forall \varepsilon > 0 : d_\infty(\{x\}, B) \leqslant \varepsilon,$$

and $d_\infty(\{x\}, B) = 0$, which is a contradiction. ∎

We conclude this section by the proof of Theorem 15.

Proof of Theorem 15. Let $R_\Delta^{\varepsilon\to 0} = \bigcap_{\varepsilon>0} \mathsf{Reach}([\![\mathcal{A}]\!]_\Delta^\varepsilon)$, for any $\Delta > 0$. If $R_{\Delta\to 0}^{\varepsilon\to 0} \cap \mathsf{Bad} = \varnothing$, since $R_{\Delta\to 0}^{\varepsilon\to 0}$ and Bad are unions of sets of the form $\{l\} \times Z_l$ where Z_l is a zone[4], Lemma 16 applies and we have $d(R_{\Delta\to 0}^{\varepsilon\to 0}, \mathsf{Bad}) > 0$. From Lemma 20, we obtain that there exists $\Delta > 0$ such that $R_\Delta^{\varepsilon\to 0} \cap \mathrm{Bad} = \varnothing$. Clearly, $R_\Delta^{\varepsilon\to 0}$ satisfies the conditions of Lemma 17, hence the existence of some Δ_1 such that $\forall \Delta \in (0, \Delta_1)$, we have $d(R_\Delta^{\varepsilon\to 0}, \mathsf{Bad}) > 0$. We pick such a $\Delta_0 \in (0, \Delta_1)$. Applying Lemma 20 to $R_{\Delta_0}^{\varepsilon\to 0}$, we get the existence of ε_0 such that $\mathsf{Reach}([\![\mathcal{A}]\!]_{\Delta_0}^{\varepsilon_0}) \cap \mathsf{Bad} = \varnothing$.

Conversely, if there exists $\Delta > 0$ and $\varepsilon > 0$ such that $\mathsf{Reach}([\![\mathcal{A}]\!]_\Delta^\varepsilon) \cap \mathrm{Bad} = \varnothing$, then trivially $R_{\Delta\to 0}^{\varepsilon\to 0} \cap \mathsf{Bad} = \varnothing$. ∎

It should be noted that a simpler proof could be achieved in the presence of only one perturbation (for guard enlargement only, such a proof can be found in [DDMR04]).

6 An algorithm for computing $R_{\Delta\to 0}^{0}$, $R_0^{\varepsilon\to 0}$ and $R_{\Delta\to 0}^{\varepsilon\to 0}$

In this section, we prove that $R_{\Delta\to 0}^{0} = R_0^{\varepsilon\to 0} = R_{\Delta\to 0}^{\varepsilon\to 0}$, and that those sets are computed by Algorithm 1 (originally proposed in [Pur98] to compute $R_0^{\varepsilon\to 0}$). The following example illustrates the algorithm, and informally justifies its correctness. The rest of this section is devoted to a formal proof of correctness.

Example. Consider the timed automaton $\mathcal{A}_\alpha$ of Fig. 1(a) (the value of α does not matter here). The reachable states of $\mathcal{A}_\alpha$ in locations ℓ_1 and ℓ_2 are depicted on Fig. 1(b) and are computed in J^* by the algorithm at line 3. Then, in the while-loop, the algorithm adds to J^* the progress cycles of the region graph of $\mathcal{A}_\alpha$ that “touch” the set J^*, and performs a reachability analysis from the new states in the classical semantics. In the example, the progress cycle (ℓ_1, R_1), (ℓ_1, R_2), (ℓ_1, R_3), (ℓ_2, R_4) shown

Algorithm 1: Algorithm for computing the limit sets $R^{\varepsilon\to 0}_{\Delta\to 0}$, $R^{0}_{\Delta\to 0}$ and $R^{\varepsilon\to 0}_{0}$ of a closed timed automaton $\mathcal{A}$.

Data: A timed automaton $\mathcal{A} = \langle \mathsf{Loc}, \mathsf{Var}, q_0, \mathsf{Lab}, \mathsf{Edg} \rangle$.
Result: The set J^*, which we will prove equals $R^{\varepsilon\to 0}_{\Delta\to 0}$, $R^{0}_{\Delta\to 0}$ and $R^{\varepsilon\to 0}_{0}$
begin
1 Construct the region graph $G = (R_{\mathcal{A}}, \to_G)$ of $\mathcal{A}$;
2 Compute the set $\mathsf{C}(G)$ of simple cycles of G ;
3 $J^* \leftarrow \mathsf{Reach}(G, [q_0])$;
4 **while** for some $p = p_0\ p_1\ \dots\ p_k \in \mathsf{C}(G)$, $[p_0] \not\subseteq J^*$ and $J^* \cap [p_0] \neq \varnothing$ **do**
5 $J^* \leftarrow J^* \cup [p_0]$;
6 $J^* \leftarrow \mathsf{Reach}(G, J^*)$;
7 **return** J^* ;
end

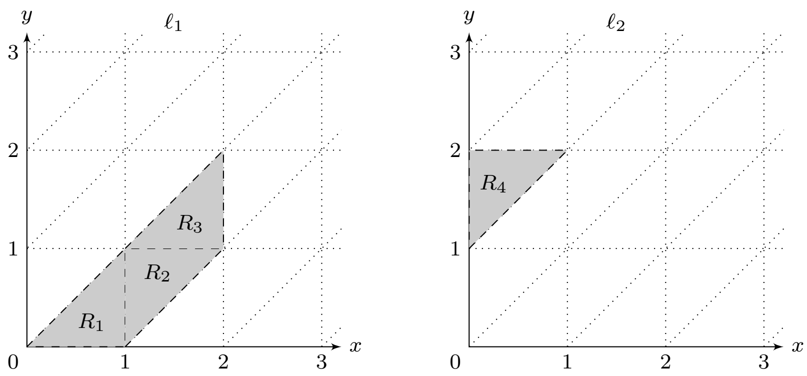


Fig. 2 A progress cycle R_1, R_2, R_3, R_4 in the region graph of $\mathcal{A}_\alpha$ of Fig. 1(a).

in Fig. 2 is added, and the set J^* computed by the algorithm is the set $R^{0}_{\Delta\to 0}$ shown in Fig. 1(c).

Observe that the algorithm manipulates only *closed* regions. In particular, the set J^* is closed by Lemma 6. The condition $J^* \cap [p_0] \neq \varnothing$ at line 4 implies that some points of J^* belong to the frontier of p_0. It is clear that in the perturbed semantics of timed automata, no matter the values of $\varepsilon, \Delta > 0$, some states of p_0 are reachable. Moreover, as we show in the sequel, *every* state of p_0 can actually be reached from every state of $[p_0]$ (and thus from $J^* \cap [p_0]$), by repeating the cycle $p = p_0\ p_1\ \dots\ p_k$ sufficiently many times (the number of iterations increases as ε and Δ tend to 0).

The main technique to prove that a state $u \in [p_0]$ can reach a state $v \in [p_0]$ in $[\![\mathcal{A}]\!]^0_\Delta$ and in $[\![\mathcal{A}]\!]^0_\varepsilon$ is the following. Assume that there is a trajectory in $[\![\mathcal{A}]\!]^0_0$ that starts and ends in u and follows the cycle p, called a *limit cycle*. We modify this trajectory using the perturbation ε or Δ to reach a state u_1 in the neighborhood of u. This is done in Theorem 29 for $[\![\mathcal{A}]\!]^0_\Delta$ and in Theorem 31 for $[\![\mathcal{A}]\!]^0_\varepsilon$. By repeating this, we construct $u_2, \dots u_k = v$ such that u_i is reachable from u_{i-1} in $[\![\mathcal{A}]\!]^0_\Delta$ (Theorem 28) and in $[\![\mathcal{A}]\!]^0_\varepsilon$ (Theorem 30). Unfortunately, we cannot assume in general that $u \in [p_0]$ has a limit cycle in $[\![\mathcal{A}]\!]^0_0$, even though p_0 has a cycle in the region graph of $\mathcal{A}$ [Pur98]. We are saved by Theorem 23 which shows that (i) from all states $x \in [p_0]$, there is a trajectory

[4] Assuming Algorithm 1 is correct, the set $J^* = R^{\varepsilon\to 0}_{\Delta\to 0}$ it computes is a union of closed regions.

in $[\![\mathcal{A}]\!]_0^0$ that reaches a state $u \in [p_0]$ that has a limit cycle, and dually (ii) all states $y \in [p_0]$ are reachable in $[\![\mathcal{A}]\!]_0^0$ from some state $v \in [p_0]$ that has a limit cycle. Notice that Theorem 23 does not involve perturbations.

6.1 Limit cycles

Definition 21 (Limit Cycle) *A* limit cycle *of a timed automaton $\mathcal{A}$ is a finite trajectory π of $[\![\mathcal{A}]\!]$ that contains at least one discrete transition and such that $\mathsf{last}(\pi) = \mathsf{first}(\pi)$.*

As suggested in [Pur98], not all states of a progress cycle have a limit cycle.

Definition 22 *Let $p = p_0\, p_1\, \dots\, p_N$ be a cycle in the region graph of a timed automaton $\mathcal{A}$ (i.e., $p_N = p_0$). For $Q_0 \subseteq [p_0]$, define the* return map *$R_p(Q_0)$ as follows:*

$$R_p(Q_0) = \left\{ q \in [p_0] \;\middle|\; \begin{array}{l} \textit{there exists a trajectory } \pi \textit{ of } [\![\mathcal{A}]\!] \textit{ that follows } p \\ \textit{such that } \mathsf{first}(\pi) \in Q_0 \textit{ and } \mathsf{last}(\pi) = q \end{array} \right\}$$

For $i \geqslant 2$, define recursively $R_p^i(Q_0) = R_p(R_p^{i-1}(Q_0))$ and let $L_{i,p}$ be the set of states that can return back to themselves after i cycles through p: $L_{i,p} = \{q \mid q \in R_p^i(\{q\})\}$. We write $L_p = \cup_{i \in \mathbb{N}_{>0}} L_{i,p}$ the set of states having a limit cycle.

The following key property of L_p is central to the proof of correctness of Algorithm 1. It states that L_p is both forward and backward reachable from all valuations in a cycle p.

Theorem 23 ([Pur98, Lemma 7.10]) *Let $p = p_0 \dots p_N$ be a cycle in the region graph of a timed automaton. For all $z \in [p_0]$, there exists z' and z'' in L_p, and trajectories π and π' in $[\![\mathcal{A}]\!]$, such that:*

- $\mathsf{first}(\pi) = z$ *and* $\mathsf{last}(\pi) = z'$ *and*
- $\mathsf{first}(\pi') = z''$ *and* $\mathsf{last}(\pi') = z$.

The proof proposed by Puri is quite sketchy. We develop here a full proof of this result, following the same steps.

Lemma 24 ([Pur98, Lemma 7.1]) *Let $p = p_0\, p_1\, \dots\, p_N$ be a path in the region graph of a timed automaton $\mathcal{A}$, let π and π' be two trajectories of $[\![\mathcal{A}]\!]$ that follow p. Then for all $\lambda \in [0, 1]$, there exists a trajectory π'' of $[\![\mathcal{A}]\!]$ that follows p and such that $\mathsf{first}(\pi'') = \lambda.\mathsf{first}(\pi) + (1 - \lambda).\mathsf{first}(\pi')$ and $\mathsf{last}(\pi'') = \lambda.\mathsf{last}(\pi) + (1 - \lambda).\mathsf{last}(\pi')$.*

Proof. Let $\pi = (q_0, t_0)\, \sigma_1\, (q_1, t_1)\, \sigma_2 \cdots \sigma_N\, (q_N, t_N)$ and $\pi' = (q'_0, t'_0)\, \sigma'_1 \cdots \sigma'_N\, (q'_N, t'_N)$. Consider the sequence

$$\pi'' = (q''_0, t''_0)\, \sigma''_1\, (q''_1, t''_1)\, \sigma''_2 \cdots \sigma''_N\, (q''_N, t''_N)$$

where for all $0 \leqslant i \leqslant N$, $q''_i = \lambda.q_i + (1 - \lambda).q'_i$ and $t''_i = \lambda.t_i + (1 - \lambda).t'_i$ and for all $1 \leqslant i \leqslant N$, $\sigma''_i = \lambda.\sigma_i + (1 - \lambda).\sigma'_i$ if $\sigma_i \in \mathbb{R}_{\geqslant 0}$ and $\sigma''_i = \sigma_i$ otherwise. It is easy to show that π'' is a trajectory in $[\![\mathcal{A}]\!]$ since regions are convex sets. ∎

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

Proof. Let $x, y \in L_p$, and $\lambda \in [0,1]$. There exists natural numbers k and l such that $x \in L_{k,p}$ and $y \in L_{l,p}$. Then $x, y \in L_{k.l,p}$, and according to Lemma 24, we have $\lambda.x + (1-\lambda).y \in L_{k \cdot l,p} \subseteq L_p$. ∎

Definition 26 *Let $p = p_0 p_1 \dots p_N$ be a cycle in the region graph of a timed automaton (i.e. $p_0 = p_N$). The* orbit graph *of p is the graph $\Theta_p = (V_\Theta, \rightarrow_\Theta)$ such that $V_\Theta = S(p_0)$ is the set of vertices of p_0 and for all $v, w \in V_\Theta$, $v \rightarrow_\Theta w$ iff $w \in R_p(\{v\})$. For $m \in \mathbb{N}$ and $v \in V_\Theta$, we define*

$$\mathsf{Succ}^m(v) = \{w \in V_\Theta \mid v \rightarrow_\Theta^m w\} \quad \text{and} \quad \mathsf{Pred}^m(v) = \{w \in V_\Theta \mid w \rightarrow_\Theta^m v\}.$$

Given a vertex $v \in V_\Theta$, the set $R_p(\{v\})$ is a closed region according to Lemma 6, and thus we have $R_p(\{v\}) = \mathsf{Conv}(\{w \in V_\Theta \mid v \rightarrow_\Theta w\})$ as a closed region contains all its vertices. More generally, we have $R_p^k(\{v\}) = \mathsf{Conv}(\{w \in V_\Theta \mid v \rightarrow_\Theta^k w\})$ for all $k \geqslant 1$.

Lemma 27 ([Pur98, Lemma 7.4**])** *Let $p = p_0 \dots p_k$ be a path in the region graph of a timed automaton $\mathcal{A}$. For all vertices $v \in S(p_0)$ of p_0, there exists a vertex $v' \in S(p_k)$ of p_k such that there exists a trajectory of $[\![\mathcal{A}]\!]$ from v to v', and conversely, for all vertices $v' \in S(p_k)$ of p_k, there exists a vertex $v \in S(p_0)$ of p_0 such that there exists a trajectory of $[\![\mathcal{A}]\!]$ from v to v'.*

Proof. Let v be a vertex of $[p_0]$. Then $\{v\}$ is a subregion of $[p-0]$, and the set of its successors in $[p_k]$ is a closed subregion of $[p_k]$. According to Lemma 14, a subregion of $[p_k]$ necessarily contains a vertex.

The same argument can be applied backwardly, since the predecessor of a subregion of p_k is a closed subregion of p_0. ∎

Proof of Theorem 23. Let $\Theta_p = (V_\Theta, \rightarrow_\Theta)$ be the orbit graph of p. Let $V = \{v \in V_\Theta \mid \exists m \in \mathbb{N}.\ v \in \mathsf{Succ}^m(v)\}$. Lemma 27 entails that every vertex in the orbit graph has an outgoing edge. Thus for all $v \in V_\Theta$, there exists an integer m_v such that for any $m \geqslant m_v$, the intersection $\mathsf{Succ}^m(v) \cap V$ is non-empty, because V_Θ is finite. Let $M = \max\{m_v \mid v \in V_\Theta\}$ be the largest such m_v. Then $\mathsf{Succ}^M(v) \cap V \neq \varnothing$ for all v. A similar argument proves the existence of M' such that $\mathsf{Pred}^{M'}(v) \cap V \neq \varnothing$ for all v.

Since $z \in [p_0]$, we can 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 element of $\mathsf{Succ}^M(v_i) \cap V$. From Lemma 24, there is a path from z to $z' = \sum_i \lambda_i w_i$ and $z' \in \mathsf{Conv}(V)$. By Lemma 25 we have $\mathsf{Conv}(V) \subseteq L$ and thus $z' \in L$.

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

6.2 Soundness of Algorithm 1: $J^* \subseteq R^0_{\Delta \to 0}$ and $J^* \subseteq R_0^{\varepsilon \to 0}$

We show that the set J^* computed by Algorithm 1 is reachable in the limit sets $R^0_{\Delta \to 0}$ and $R_0^{\varepsilon \to 0}$. In particular, for all progress cycles that are added to J^* by the algorithm, we show that every point of the cycle is reachable if either a drift on clocks or an enlargement of the guards is allowed, no matter how small it is.

The proof is based on Theorem 23 and on the fact that for all progress cycles p, the set L_p is a strongly connected component of both $[\![\mathcal{A}]\!]^0_\Delta$ and $[\![\mathcal{A}]\!]^\varepsilon_0$ for all $\Delta, \varepsilon > 0$. Hence in L_p, every state is reachable from every state for the perturbed semantics, and thus similarly, in each region of the cycle p every state is reachable from every state by Theorem 23.

Apart from Theorem 39, all the results of this section apply to progress cycles but do *not* require Assumption 9.

Imprecise guards: $J^* \subseteq R^0_{\Delta\to 0}$.

Theorem 28 *Let $\mathcal{A}$ be a timed automaton, let $p = p_0\, p_1 \cdots p_N$ be a progress cycle of the region graph of $\mathcal{A}$, and $\Delta \in \mathbb{R}_{>0}$. For all states $u, v \in L_p$, there exists a trajectory π of $[\![\mathcal{A}]\!]^0_\Delta$ such that $\mathsf{first}(\pi) = u$ and $\mathsf{last}(\pi) = v$.*

This theorem results immediately from the following lemma.

Lemma 29 *Let $\mathcal{A}$ be a timed automaton, let $p = p_0\, p_1 \cdots p_N$ be a progress cycle of the region graph of $\mathcal{A}$. For all $\Delta \in \mathbb{R}_{>0}$, for all state $u \in L_p$ and for all neighbour state $v \in [p_0] \cap \mathcal{N}_\infty(u, \frac{\Delta}{2})$, there exists a trajectory π' of $[\![\mathcal{A}]\!]^0_\Delta$ such that $\mathsf{first}(\pi') = u$ and $\mathsf{last}(\pi') = v$.*

Proof. Let $\mathcal{A} = \langle \mathsf{Loc}, \mathsf{Var}, q_0, \mathsf{Lab}, \mathsf{Edg}\rangle$. Since $u \in L_p$, there exists a trajectory π of $[\![\mathcal{A}]\!]^0_0$ that follows p a certain number of times and such that $\mathsf{first}(\pi) = \mathsf{last}(\pi) = u$. We slightly modify π to make it stutter-free (we insert a zero length timed transition between two consecutive discrete transitions, and we merge consecutive timed transitions). Assume that[5]:

$$\begin{aligned}\pi = (\ell_0, u_0) &\xrightarrow{t_0} (\ell_0, u_0') \xrightarrow[R_0]{\sigma_0} (\ell_1, u_1) \xrightarrow{t_1} (\ell_1, u_1') \xrightarrow[R_1]{\sigma_1} \cdots \\ \cdots &\xrightarrow[R_{m-2}]{\sigma_{m-2}} (\ell_{m-1}, u_{m-1}) \xrightarrow{t_{m-1}} (\ell_{m-1}, u_{m-1}') \xrightarrow[R_{m-1}]{\sigma_{m-1}} (\ell_m, u_m) \xrightarrow{t_m} (\ell_m, u_m')\end{aligned}$$

with $u_0 = u_m' = u$. Each $t_i \in \mathbb{R}_{\geqslant 0}$ and $\sigma_i \in \Sigma$. We annotate π with sets of clocks $R_i \subseteq \mathsf{Var}$ that are reset by discrete transitions σ_i. Note that $\bigcup_{i=0}^{i=m-1} R_i = \mathsf{Var}$ by Assumption 9.

Intuitively, we prove the lemma by modifying the length of the timed transitions of π so that the clocks are reset slightly earlier or later than in π. We obtain a trajectory of $[\![\mathcal{A}]\!]^0_\Delta$ because the guards are enlarged and therefore they are enabled in the states of the new trajectory.

Let the representation of p_0 be given by $(a_x)_{x\in\mathsf{Var}}$ and $(X_i)_{0\leqslant i\leqslant k}$. For all valuations $w \in p_0$, we have:

- for all $x \in \mathsf{Var}$: $\lfloor w(x) \rfloor = a_x$;
- for all $x \in X_0$: $\langle w(x)\rangle = 0$;
- for all i and for all $x, y \in X_i$: $\langle w(x)\rangle = \langle w(y)\rangle$;
- for all $i < j$ and for all $x \in X_i,\ y \in X_j$: $\langle w(x)\rangle < \langle w(y)\rangle$;

[5] It is not restrictive to assume that π starts and ends with a timed transition as timed transitions of length zero are allowed.

By hypothesis, p is a progress cycle and each clock is reset at least once along π. For each clock $x \in \mathsf{Var}$, let α_x be the index of the last transition of π in which x is reset. Formally, we have:

$$x \in R_{\alpha_x} \qquad\qquad \forall i > \alpha_x : x \notin R_i \tag{2}$$

Then, for each clock $x \in \mathsf{Var}$, we have:

$$u_0(x) = u'_m(x) = u(x) = \sum_{i=\alpha_x+1}^{m} t_i \tag{3}$$

Let $v \in [p_0] \cap \mathcal{N}_\infty(u, \frac{\Delta}{2})$ and for each $x \in \mathsf{Var}$ let $\delta_x = v(x) - u(x)$. Clearly, we have $|\delta_x| \leqslant \frac{\Delta}{2}$. Moreover since $v \in [p_0]$, the closed version of the above inequalities defining p_0 are satisfied by v. Let $\langle\langle v(x)\rangle\rangle = v(x) - a_x$, we have:

- for all $x \in \mathsf{Var}$: $0 \leqslant \langle\langle v(x)\rangle\rangle \leqslant 1$;
- for all $x \in X_0$: $\langle\langle v(x)\rangle\rangle = 0$;
- for all i and for all $x, y \in X_i$: $\langle\langle v(x)\rangle\rangle = \langle\langle v(y)\rangle\rangle$;
- for all $i < j$ and for all $x \in X_i,\ y \in X_j$: $\langle\langle v(x)\rangle\rangle \leqslant \langle\langle v(y)\rangle\rangle$;

This entails that:

- for all i and for all $x, y \in X_i$: $\delta_x = \langle\langle v(x)\rangle\rangle - \langle u(x)\rangle = \langle\langle v(y)\rangle\rangle - \langle u(y)\rangle = \delta_y$;
- for all $x, y \in \mathsf{Var}$ such that $u(x) < u(y)$ (and hence $\alpha_x > \alpha_y$ from Eq. (3)), we have $v(x) \leqslant v(y)$ and thus $u(x) + \delta_x \leqslant u(y) + \delta_y$, that is:

$$\delta_x - \delta_y \leqslant u(y) - u(x) = \sum_{i=\alpha_y+1}^{\alpha_x} t_i \tag{4}$$

Let $\Gamma = \{\alpha_x \mid x \in \mathsf{Var}\} = \{\alpha_1, \ldots, \alpha_l\}$ be the set of positions in π where a clock is reset for the last time. Assume without loss of generality that $\alpha_1 < \alpha_2 < \cdots < \alpha_l$ and that for all $1 \leqslant i \leqslant l$, the clock $x_i \in \mathsf{Var}$ is such that that $\alpha_{x_i} = \alpha_i$. Consider the time stamps in π as the following block-sequence, and construct the sequence $(t'_i)_{0 \leqslant i \leqslant m}$ by adding a *shift* given as follows:

$$\begin{array}{c}
[t_0 \ldots t_{\alpha_1}][t_{\alpha_1+1} \ldots t_{\alpha_2}] \ldots [t_{\alpha_{j-1}+1} \ldots t_{\alpha_j}] \ldots [t_{\alpha_{l-1}+1} \ldots t_{\alpha_l}][t_{\alpha_l+1} \ldots t_m] \\
+0 \qquad +\delta_1 - \delta_2 \quad \ldots \quad +\delta_{j-1} - \delta_j \quad \ldots \quad +\delta_{l-1} - \delta_l \qquad +\delta_l \\
\hline
= [t'_0 \ldots t'_{\alpha_1}][t'_{\alpha_1+1} \ldots t'_{\alpha_2}] \ldots [t'_{\alpha_{j-1}+1} \ldots t'_{\alpha_j}] \ldots [t'_{\alpha_{l-1}+1} \ldots t'_{\alpha_l}][t'_{\alpha_l+1} \ldots t'_m]
\end{array}$$

where each t'_i is obtained from t_i by distributing the shift of each block over the time stamps of the block. This can be done in such a way that each t'_i is nonnegative for all $0 \leqslant i \leqslant m$ since for all $i \leqslant \alpha_1$ we have $t'_i = t_i$, for all $2 \leqslant j \leqslant l$ we have:

$$\sum_{i=\alpha_{j-1}+1}^{\alpha_j} t'_i = \left(\sum_{i=\alpha_{j-1}+1}^{\alpha_j} t_i \right) + \delta_{j-1} - \delta_j \geqslant 0 \qquad \text{(by Eq. (4))}$$

and finally for all $i \geqslant \alpha_l + 1$ we have:

$$\sum_{i=\alpha_l+1}^{m} t'_i \;=\; \sum_{i=\alpha_l+1}^{m} t_i + \delta_l = u(x_l) + \delta_l = v(x_l) \geqslant 0.$$

We now construct the trajectory π' from π by replacing each t_i by t'_i:

$$\pi' = (\ell_0, v_0) \xrightarrow{t'_0} (\ell_0, v'_0) \xrightarrow[R_0]{\sigma_0} (\ell_1, v_1) \xrightarrow{t'_1} (\ell_1, v'_1) \xrightarrow[R_1]{\sigma_1} \cdots$$
$$\cdots \xrightarrow[R_{m-2}]{\sigma_{m-2}} (\ell_{m-1}, v_{m-1}) \xrightarrow{t'_{m-1}} (\ell_{m-1}, v'_{m-1}) \xrightarrow[R_{m-1}]{\sigma_{m-1}} (\ell_m, v_m) \xrightarrow{t'_m} (\ell_m, v'_m)$$

where $v_0 = u_0 = u$, for all $0 \leqslant i \leqslant m : v'_i = v_i + t'_i$ and for all $1 \leqslant i \leqslant m : v_i = v'_{i-1}[R_{i-1} := 0]$. We claim that π' is a trajectory of $[\![\mathcal{A}]\!]^0_\Delta$. To show this, we must verify that the guard (which is enlarged by Δ in $[\![\mathcal{A}]\!]^0_\Delta$) of each discrete transition σ_i is satisfied by v'_i. Since π is a trajectory of $[\![\mathcal{A}]\!]$, we know that each u'_i satisfies the corresponding guard under the classical semantics. Therefore, it is sufficient to prove that the difference $|u'_i(x) - v'_i(x)|$ is bounded by Δ for all $x \in \mathsf{Var}$. To do that, let j be the greatest index such that $j \leqslant i$ and $u_j(x) = v_j(x)$ (such an index exists because $u_0 = v_0$). Clearly, the difference $|u'_i(x) - v'_i(x)|$ is bounded by the sum of the shifts that we have introduced between index j and i. For uniformity, let the first shift be $\delta_0 - \delta_1$ with $\delta_0 = \delta_1$ and the last shift be $\delta_l - \delta_{l+1}$ with $\delta_{l+1} = 0$. Notice that $|\delta_i| \leqslant \frac{\Delta}{2}$ holds for all $i = 0, \ldots, l+1$. If a block $\left[t_{\alpha_{p-1}+1} \ldots t_{\alpha_p}\right]$ is such that $j \leqslant \alpha_{p-1} + 1$ and $\alpha_p \leqslant i$, then the whole shift $\delta_{p-1} - \delta_p$ counts in the sum. On the other hand, if i or j lies inside the block, then only a portion $\alpha(\delta_{p-1} - \delta_p)$ of the shift counts where $\alpha \in [0, 1]$. Accordingly, the sum of the shifts can take one of the three forms (where $\alpha, \beta \in [0, 1]$):

- $s_1 = \alpha(\delta_p - \delta_{p+1})$ (if i and j lie in the same block)
- $s_2 = \alpha(\delta_p - \delta_{p+1}) + \beta(\delta_{p+1} - \delta_{p+2})$ (if i and j lie in consecutive blocks)
- $s_3 = \alpha(\delta_p - \delta_{p+1}) + \delta_{p+1} - \delta_q + \beta(\delta_q - \delta_{q+1})$ (otherwise)

It is easy to show the following bounds:

- $|s_1| \leqslant \alpha \cdot 2 \cdot \frac{\Delta}{2} \leqslant \Delta$
- $\left.\begin{array}{l} |s_2| = |\alpha(\delta_p - \delta_{p+2}) + (\beta - \alpha)(\delta_{p+1} - \delta_{p+2})| \leqslant \alpha \cdot \Delta + |\beta - \alpha|\Delta \\ |s_2| = |(\alpha - \beta)(\delta_p - \delta_{p+1}) + \beta(\delta_p - \delta_{p+2})| \leqslant |\alpha - \beta|\Delta + \beta \cdot \Delta \end{array}\right\} \Rightarrow |s_2| \leqslant \Delta$
- $\left.\begin{array}{l} |s_3| = |\alpha(\delta_p - \delta_{q+1}) + (1 - \beta)(\delta_{p+1} - \delta_q) + (\beta - \alpha)(\delta_{p+1} - \delta_{q+1})| \\ |s_3| = |(\alpha - \beta)(\delta_p - \delta_q) + (1 - \alpha)(\delta_{p+1} - \delta_q) + \beta(\delta_p - \delta_{q+1})| \end{array}\right\} \Rightarrow |s_3| \leqslant \Delta$

which shows that $|u'_i(x) - v'_i(x)| \leqslant \Delta$ for all $x \in \mathsf{Var}$.

Finally, since the sets of clocks R_i that are reset in π' are the same as in π, Eq. (3) applies and we get for all $x \in \mathsf{Var}$:

$$v'_m(x) \;=\; \sum_{i=\alpha_x+1}^{m} t'_i \;=\; \sum_{i=\alpha_x+1}^{m} t_i + \delta_x \;=\; u'_m(x) + \delta_x = u(x) + \delta_x$$

Hence $v'_m = v$. It follows that $\mathsf{first}(\pi') = u$ and $\mathsf{last}(\pi') = v$ as required. ∎

Drifting clocks: $J^* \subseteq R_0^{\varepsilon \to 0}$.

We prove a result similar to Theorem 28 for drifting clocks.

Theorem 30 (See also [Pur98, Lemma 7.11]) *Let $\mathcal{A}$ be a timed automaton, let $p = p_0\, p_1 \cdots p_N$ be a progress cycle of the region graph of $\mathcal{A}$, and $\varepsilon \in \mathbb{R}_{>0}$. For all states $u, v \in L_p$, there exists a trajectory π of $[\![\mathcal{A}]\!]_0^\varepsilon$ such that $\mathsf{first}(\pi) = u$ and $\mathsf{last}(\pi) = v$.*

The proof of Theorem 30 relies on the following lemma, implying that it is possible to go from a state $x_0 = u \in L_p$ to a state $x_k = v \in L_p$ in $[\![\mathcal{A}]\!]_0^\varepsilon$ by successively reaching $x_1, x_2, \dots x_k$ where x_{i+1} is at bounded distance from x_i. The key is that the bound depends only on the extremal states u and v.

Lemma 31 *Let $\mathcal{A}$ be a timed automaton and $p = p_0\, p_1 \cdots p_N$ be a progress cycle of the region graph of $\mathcal{A}$. For all $u, v \in L_p$, for all $\varepsilon \in \mathbb{R}_{>0}$, there exists $\delta > 0$ such that for all $x \in \mathit{Conv}(\{u, v\})$ and for all $y \in L_p \cap \mathcal{N}_\infty(x, \delta)$, there exists a trajectory π in $[\![\mathcal{A}]\!]_0^\varepsilon$ such that $\mathsf{first}(\pi) = x$ and $\mathsf{last}(\pi) = y$.*

To prove Lemma 31, we apply a drift on a limit cycle trajectory starting in u. Since the effect of drifts is proportional to the duration of the trajectory, we need a bound on the minimal duration of such a trajectory, in order to guarantee a lower bound on the perturbation that we can enforce. This is done in Lemma 32. Then, to show that every state in $\mathcal{N}_\infty(x, \eta)$ is reachable from a state x, we need to establish the relationship between numbers δ and η in the following statement:

$$\forall x' \in [r'] \cap \mathcal{N}_\infty(x_{i+1}, \delta) \cdot \exists x \in [r] \cap \mathcal{N}_\infty(x_i, \eta) : x \xrightarrow{\tau} x' \text{ in } [\![\mathcal{A}]\!]_0^\varepsilon \text{ for some } \tau \in \mathbb{R}_{\geqslant 0}.$$

This statement can be used inductively to show that a neighborhood of $x_k = v$ (and thus v itself) can be reached from $x_0 = u$, as in Theorem 30. When $\varepsilon = 0$, we show in Lemma 33 that $\eta = 2\delta$ fits. To have a proof of Lemma 31, we would need that $\eta = 0$ while $\delta > 0$. To obtain this, we use the drifts on clocks. The proof is quite involved because Lemma 33 tends to require a larger value of η to guarantee some δ, while the drifts help us to decrease η.

In the next lemma, we show that every limit-cycle trajectory of strictly positive duration can be extended to a limit-cycle trajectory of duration at least $\frac{1}{2}$ without increasing the size of π more than twice. This last condition is important as otherwise, the lemma would be trivially true.

Lemma 32 *Let $\mathcal{A}$ be a timed automaton, let $p = p_0\, p_1\, p_2 \dots p_N$ be a progress cycle in the region graph of $\mathcal{A}$. If there exists a limit cycle π of $[\![\mathcal{A}]\!]$ that follows p such that $\mathsf{Duration}(\pi) > 0$, then there exists a limit cycle π' of $[\![\mathcal{A}]\!]$ with $\mathsf{first}(\pi') = \mathsf{first}(\pi)$ and such that $|\pi'| \leqslant 2|\pi|$ and $\mathsf{Duration}(\pi') \geqslant 1/2$.*

Proof. The result is immediate if $\mathsf{Duration}(\pi) \geqslant 1/2$. Assume $\mathsf{Duration}(\pi) < 1/2$, and let $k = |\pi|$ and $u = \mathsf{first}(\pi) = \mathsf{last}(\pi)$. Let π_2 be the trajectory obtained by repeating π twice. We have $|\pi_2| = 2k$ and $\mathsf{first}(\pi_2) = \mathsf{state}_k(\pi_2) = \mathsf{last}(\pi_2) = u$. Since all clocks are reset along π, their value remain strictly less than $1/2$ in every state of π and π_2. By the fact that $\mathsf{Duration}(\pi) > 0$, there must be at least one timed transition in π with a strictly positive time stamp. Consider the first such transition in π_2, and let increase its length by $1/2$ time units, yielding a new trajectory π' in which each clock remains below 1. Therefore, the same transitions as in π_2 can be taken as the guards satisfied by a state of π are also satisfied by the corresponding state in π'. Observe that we keep in the second half of the trajectory π' the same sequence of transitions as in π, and since all clocks are reset along π, we obtain $\mathsf{last}(\pi') = u = \mathsf{first}(\pi')$, $|\pi'| = 2|\pi|$ and $\mathsf{Duration}(\pi') \geqslant 1/2$. ■

Lemma 33 *Let $\mathcal{A}$ be a timed automaton, let r and r' be two regions of $\mathcal{A}$ s.t. $(\ell, r) \xrightarrow{\mathsf{time}} (\ell, r')$ in the region graph of $\mathcal{A}$. For all $u \in r$, $v \in r'$ and $\tau \in \mathbb{R}_{>0}$ such that $(\ell, u) \xrightarrow{\tau} (\ell, v)$ in $[\![\mathcal{A}]\!]_0^0$, and for all $\delta \geqslant 0$, for all $y \in \mathcal{N}_\infty(v, \delta) \cap [r']$, there exists $x \in \mathcal{N}_\infty(u, 2\delta) \cap [r]$ and $\tau' \in \mathbb{R}_{\geqslant 0}$ such that $(\ell, x) \xrightarrow{\tau'} (\ell, y)$ in $[\![\mathcal{A}]\!]_0^0$.*

Proof. Let n be the number of clocks of $\mathcal{A}$. Reminiscent of the normal form DBM representation of regions, let $\alpha_i, \beta_i, m_{i,j} \in \mathbb{Z}$ and $\alpha'_i, \beta'_i, m'_{i,j} \in \mathbb{Z}$ be the tightest constants such that for all valuations u, v, we have $u \in [r]$ and $v \in [r']$ if and only if for all $1 \leqslant i, j \leqslant n$:

$$u_i - u_j \leqslant m_{i,j} \qquad \alpha_i \leqslant u_i \leqslant \beta_i \qquad ([r])$$
$$v_i - v_j \leqslant m'_{i,j} \qquad \alpha'_i \leqslant v_i \leqslant \beta'_i \qquad ([r'])$$

In particular, this entails that $-m_{j,i} \leqslant u_i - u_j \leqslant \beta_i - \alpha_j$ and $-m'_{j,i} \leqslant v_i - v_j \leqslant \beta'_i - \alpha'_j$ for all $1 \leqslant i, j \leqslant n$, and since the constants are tight:

$$-m_{j,i} \leqslant m_{i,j} \leqslant \beta_i - \alpha_j \qquad -m'_{j,i} \leqslant m'_{i,j} \leqslant \beta'_i - \alpha'_j \qquad (5)$$

Now, let $u \in r$, $v \in r'$ and $\tau \in \mathbb{R}_{>0}$ such that $(\ell, u) \xrightarrow{\tau} (\ell, v)$ in $[\![\mathcal{A}]\!]_0^0$, and let $\delta \geqslant 0$ and $y \in \mathcal{N}_\infty(v, \delta) \cap [r']$. Since r' is a time successor of r and $v = u + \tau$, we have for all $1 \leqslant i, j \leqslant n$:

$$m_{i,j} = m'_{i,j} \qquad v_i - v_j = u_i - u_j \qquad (6)$$

We define the valuation $D = y - v$. Since $y \in \mathcal{N}_\infty(v, \delta)$, we have $\|D\|_\infty \leqslant \delta$ and since $y \in [r']$, we have for all $1 \leqslant i, j \leqslant n$:

$$(v_i + D_i) - (v_j + D_j) \leqslant m'_{i,j} \qquad \alpha'_i \leqslant v_i + D_i \leqslant \beta'_i \qquad (7)$$

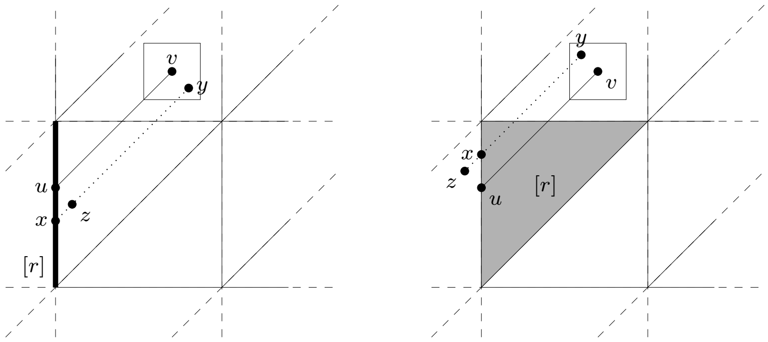
Now let $z = u + D$. As shown on Fig. 3, we might have $z \notin [r]$. Thus, we have to construct a neighbour x of z that belongs to $[r]$ and such that $x \xrightarrow{\tau'} y$ for some $\tau' \in \mathbb{R}_{\geqslant 0}$. By Eq. (6) and (7), we have for all $1 \leqslant i, j \leqslant n$:

$$z_i - z_j = (u_i + D_i) - (u_j + D_j) = (v_i + D_i) - (v_j + D_j) \leqslant m_{i,j} \qquad (8)$$

1 First, assume that for some i_0, we have $\alpha_{i_0} = \beta_{i_0}$: This means that the interior of r is empty, as on Fig. 3(a). Let $t = -D_{i_0}$. Notice that the value of t is independent of the choice of i_0. Indeed, if for some $j \neq i_0$ we have $\alpha_j = \beta_j$, then using Eq. (5) we get:

$$\alpha_{i_0} - \beta_j \leqslant -m_{j,i_0} \leqslant m_{i_0,j} \leqslant \beta_{i_0} - \alpha_j$$

and $-m_{j,i_0} = m_{i_0,j}$ since $\alpha_{i_0} - \beta_j = \beta_{i_0} - \alpha_j$. By Eq. (6), we have $-m'_{j,i_0} = m'_{i_0,j}$ and thus in the region $[r']$, we have $v_{i_0} - v_j = -m'_{j,i_0} = m'_{i_0,j} = y_{i_0} - y_j$ and thus $D_{i_0} = D_j$.



(a) The interior of $[r]$ is empty.

(b) The interior of $[r]$ is not empty.

Fig. 3 Construction of a predecessor of y in the closed region $[r]$.

Now, let $x = z + t$ so that $x_{i_0} = u_{i_0}$. We show that $x \in [r]$. Clearly, by Eq. (8) we have:

$$x_i - x_j = z_i - z_j \leqslant m_{i,j}$$

And in particular, for all $1 \leqslant j \leqslant n : -m_{i_0,j} \leqslant x_j - x_{i_0} \leqslant m_{j,i_0}$. Since $x_{i_0} = u_{i_0} = \alpha_{i_0} = \beta_{i_0}$ and by Eq. (5) we have:

$$\alpha_j \leqslant \beta_{i_0} - m_{i_0,j} \leqslant x_j \leqslant m_{j,i_0} + \alpha_{i_0} \leqslant \beta_j$$

Now, we have $\|x - u\|_\infty = \|D + t\|_\infty \leqslant 2\|D\|_\infty \leqslant 2\delta$ and thus $x \in \mathcal{N}_\infty(u, 2\delta) \cap [r]$.

2 Second, assume that for all i, we have $\alpha_i < \beta_i$: This means that the interior of r is not empty, as on Fig. 3(b). We define the following sets:

$$I = \{i \mid \alpha_i > z_i\} \qquad I' = \{i \mid z_i > \beta_i\}$$

- If $I = \varnothing$ and $I' = \varnothing$, then $z \in [r]$ by Eq. (8) and we take $x = z$. We have $\|x - u\|_\infty = \|z - u\|_\infty = \|D\|_\infty \leqslant \delta$.
- If $I \neq \varnothing$, then define $t = \max\{\alpha_i - z_i \mid i \in I\}$ and let i_0 be an index in I such that $t = \alpha_{i_0} - z_{i_0}$. Clearly $t > 0$ and since $t = \alpha_{i_0} - u_{i_0} - D_{i_0}$ and $\alpha_{i_0} \leqslant u_{i_0}$, we have $t \leqslant -D_{i_0}$. We take $x = z + t$ so that $x_{i_0} = \alpha_{i_0}$. We show that $x \in [r]$. Clearly, by Eq. (8) we have:

  $$x_i - x_j = z_i - z_j \leqslant m_{i,j}$$

  In particular, for all $1 \leqslant i \leqslant n$ we have $x_j - x_{i_0} \leqslant m_{j,i_0} \leqslant \beta_j - \alpha_{i_0}$ by Eq. (5). Since $x_{i_0} = \alpha_{i_0}$, this yields $x_j \leqslant \beta_j$. Moreover, for all $i \in I$ we have $x_i = z_i + t \geqslant \alpha_i$ by definition of t, and for all $i \notin I$ we have $x_i = z_i + t \geqslant z_i \geqslant \alpha_i$. Finally, we have $\|x - u\|_\infty = \|D + t\|_\infty \leqslant 2\|D\|_\infty \leqslant 2\delta$.
- If $I' \neq \varnothing$, then define $t = \min\{\beta_i - z_i \mid i \in I'\}$ and let i_0 be an index in I' such that $t = \beta_{i_0} - z_{i_0}$. Clearly $t < 0$ and since $t = \beta_{i_0} - u_{i_0} - D_{i_0}$ and $u_{i_0} \leqslant \beta_{i_0}$, we have $t \geqslant -D_{i_0}$. We take $x = z + t$ so that $x_{i_0} = \beta_{i_0}$. We show that $x \in [r]$. Clearly, by Eq. (8) we have:

  $$x_i - x_j = z_i - z_j \leqslant m_{i,j}$$

In particular, for all $1 \leqslant j \leqslant n$ we have $x_{i_0} - x_j \leqslant m_{i_0,j} \leqslant \beta_{i_0} - \alpha_j$ by Eq. (5). Since $x_{i_0} = \beta_{i_0}$, this yields $x_j \geqslant \alpha_j$. Moreover, for all $i \in I'$ we have $x_i = z_i + t \leqslant \beta_i$ by definition of t, and for all $i \notin I'$ we have $x_i = z_i + t \leqslant z_i \leqslant \beta_i$. Finally, we have $\|x - u\|_\infty = \|D + t\|_\infty \leqslant 2\|D\|_\infty \leqslant 2\delta$.

In each case, we have $x \in \mathcal{N}_\infty(u, 2\delta) \cap [r]$ and $(\ell, x) \xrightarrow{\tau'} (\ell, y)$ in $[\![\mathcal{A}]\!]_0^0$ for $\tau' = \tau - t$ (obviously we have $\tau' \geqslant 0$ because r' is a time successor of r). ∎

Lemma 34 *Let $\mathcal{A}$ be a timed automaton. Let (ℓ, x) and (ℓ, y) be two states of $[\![\mathcal{A}]\!]$, and $\tau \in \mathbb{R}_{\geqslant 0}$ such that $(\ell, x) \xrightarrow{\tau} (\ell, y)$ in $[\![\mathcal{A}]\!]$. For all $\varepsilon \in \mathbb{R}_{>0}$, for all $x' \in \mathcal{N}_\infty(x, \varepsilon\tau)$: $(\ell, x') \xrightarrow{\tau} (\ell, y)$ in $[\![\mathcal{A}]\!]_0^\varepsilon$.*

Proof. The result is immediate if $\tau = 0$. Otherwise, it suffices to set the rate of each clock c of $\mathcal{A}$ to $1 - (x'(c) - x(c))/\tau$, which lies between $1 - \varepsilon$ and $1 + \varepsilon$. ∎

Lemma 35 *Let $\mathcal{A}$ be a timed automaton, let r and r' be two regions of $\mathcal{A}$ such that $r \to r'$ in the region graph of $\mathcal{A}$. For all $u \in [r]$, $v \in [r']$ and $\varepsilon \in \mathbb{R}_{>0}$:*

- *if there is a timed transition $u \xrightarrow{\tau} v$ in $[\![\mathcal{A}]\!]$, then for all $\eta \in \mathbb{R}_{\geqslant 0}$, we have:*

$$\forall y \in \mathcal{N}_\infty(v, \frac{\eta + \varepsilon\tau}{2 + 3\varepsilon}) \cap [r'] \cdot \ \exists x' \in \mathcal{N}_\infty(u, \eta) \cap [r] : \ x' \xrightarrow{\tau'} y \text{ in } [\![\mathcal{A}]\!]_0^\varepsilon;$$

- *if there is an action transition $u \xrightarrow{\sigma} v$ in $[\![\mathcal{A}]\!]$, then for all $\eta \in \mathbb{R}_{\geqslant 0}$, we have*

$$\forall y \in \mathcal{N}_\infty(v, \eta) \cap [r'] \cdot \ \exists x \in \mathcal{N}_\infty(u, \eta) \cap [r] : \ x \xrightarrow{\sigma} y \text{ in } [\![\mathcal{A}]\!]_0^\varepsilon.$$

Proof. We only prove the first part of the lemma, the second part being quite obvious. We have $v = u + \tau$. Let $\delta = (\eta + \varepsilon\tau)/(2 + 3\varepsilon)$ and let $y \in \mathcal{N}_\infty(v, \delta) \cap [r']$. From Lemma 33, there exists $x \in \mathcal{N}_\infty(u, 2\delta) \cap [r]$ such that $x \xrightarrow{\tau'} y$ in $[\![\mathcal{A}]\!]$ for some $\tau' \in \mathbb{R}_{\geqslant 0}$. So we have $y = x + \tau'$. Using the triangle inequalities, we have:

$$\tau' = \|y - x\|_\infty = \|(v - u) - [(x - u) + (v - y)]\|_\infty \geqslant \tau - 3\delta \tag{9}$$

Consider the set $S = \mathcal{N}_\infty(x, \varepsilon\tau') \cap \mathsf{Conv}(\{u, x\})$. Since $d_\infty(u, x) \leqslant 2\delta$, there exists $x' \in S$ such that:

$$\begin{cases} d_\infty(u, x') = 0 & \text{if } d_\infty(u, x) \leqslant \varepsilon\tau' \text{ (take } x' = u) \\ d_\infty(u, x') \leqslant 2\delta - \varepsilon\tau' & \text{if } d_\infty(u, x) > \varepsilon\tau' \end{cases}$$

Since $[r]$ is convex and $x, u \in [r]$, we have $x' \in [r]$, and since $x \xrightarrow{\tau'} y$ in $[\![\mathcal{A}]\!]$, Lemma 34 entails that $x' \xrightarrow{\tau'} y$ in $[\![\mathcal{A}]\!]_0^\varepsilon$. To complete the proof, we have to show that $x' \in \mathcal{N}_\infty(u, \eta)$, that is $d_\infty(u, x') \leqslant \eta$. Starting from Eq. (9), we have:

$$\varepsilon(\tau - \tau') \leqslant 3\varepsilon\delta = (2 + 3\varepsilon)\delta - 2\delta = \eta + \varepsilon\tau - 2\delta$$

and thus $2\delta - \varepsilon\tau' \leqslant \eta$ which entails $d_\infty(u, x') \leqslant \eta$. ∎

Lemma 36 *Let $\mathcal{A}$ be a timed automaton, let $\varepsilon \in \mathbb{R}_{>0}$ and $K_\varepsilon = 1/(2+3\varepsilon)$. Let $p = p_0 p_1 \cdots p_N$ be a path in the region graph of $\mathcal{A}$. Let π be a trajectory of $[\![\mathcal{A}]\!]$ that follows p and let $u = \mathsf{first}(\pi)$, $v = \mathsf{last}(\pi)$ and $T = \mathsf{Duration}(\pi)$. For all $y \in \mathcal{N}_\infty(v, K_\varepsilon^N \varepsilon T) \cap [p_N]$, there exists a trajectory π' in $[\![\mathcal{A}]\!]_0^\varepsilon$ that follows p and such that $\mathsf{first}(\pi') = u$ and $\mathsf{last}(\pi') = y$.*

Proof. Let $\pi = (q_0, t_0)\sigma_1(q_1, t_1)\sigma_2 \ldots \sigma_N(q_N, t_N)$ with $t_0 = 0$ and $t_N = T$. Define $\varepsilon_i = K_\varepsilon^i \varepsilon t_i$. We show that for all $0 \leqslant i < N$, for all $y \in \mathcal{N}_\infty(q_{i+1}, \varepsilon_{i+1}) \cap [r_{i+1}]$, there exists $x \in \mathcal{N}_\infty(q_i, \varepsilon_i) \cap [r_i]$ such that there exists a transition from x to y in $[\![\mathcal{A}]\!]_0^\varepsilon$:

- if $q_i \xrightarrow{\sigma_{i+1}} q_{i+1}$ is a discrete transition, then we have $\varepsilon_{i+1} \leqslant \varepsilon_i$ because $t_{i+1} = t_i$ and $K_\varepsilon \leqslant 1$. The claims follows then directly from Lemma 35.
- otherwise, we have a timed transition $q_i \xrightarrow{\tau} q_{i+1}$ and $t_{i+1} = t_i + \tau$. By Lemma 35 with $\eta = \varepsilon_i$, we have:

$$\forall y \in \mathcal{N}_\infty(q_{i+1}, K_\varepsilon(\varepsilon_i + \varepsilon\tau)) \cap [r_{i+1}] \cdot \exists x' \in \mathcal{N}_\infty(q_i, \varepsilon_i) \cap [r_i] :\ x' \xrightarrow{\tau'} y \text{ in } [\![\mathcal{A}]\!]_0^\varepsilon.$$

  Since $K_\varepsilon \leqslant 1$, we have:

$$K_\varepsilon(\varepsilon_i + \varepsilon\tau) = K_\varepsilon^{i+1}\varepsilon t_i + K_\varepsilon \varepsilon\tau \geqslant K_\varepsilon^{i+1}\varepsilon(t_i + \tau) = K_\varepsilon^{i+1}\varepsilon t_{i+1} = \varepsilon_{i+1}$$

  Hence, $\mathcal{N}_\infty(q_{i+1}, \varepsilon_{i+1}) \subseteq \mathcal{N}_\infty(q_{i+1}, K_\varepsilon(\varepsilon_i + \varepsilon\tau))$ and we have:

$$\forall y \in \mathcal{N}_\infty(q_{i+1}, \varepsilon_{i+1}) \cap [r_{i+1}].\ \exists x' \in \mathcal{N}_\infty(q_i, \varepsilon_i) \cap [r_i].\ x' \xrightarrow{\tau'} y \text{ in } [\![\mathcal{A}]\!]_0^\varepsilon.$$

Applying this result for each $0 \leqslant i < N$, we obtain immediately that for all $y \in \mathcal{N}_\infty(q_N, \varepsilon_N) \cap [r_N]$, there exists $x \in \mathcal{N}_\infty(q_0, \varepsilon_0) \cap [r_0]$ such that there exists a trajectory π' in $[\![\mathcal{A}]\!]_0^\varepsilon$ that follows p with $\mathsf{first}(\pi') = x$ and $\mathsf{last}(\pi') = y$. Finally, we have $q_N = \mathsf{last}(\pi)$ and $q_0 = \mathsf{first}(\pi)$ so that $x = u$ since $\varepsilon_0 = 0$ and $\mathcal{N}_\infty(q_0, 0) = \{q_0\}$. ■

Lemma 37 *Let $\mathcal{A}$ be a timed automaton and p be a progress cycle of the region graph of $\mathcal{A}$. For all $u, v \in L_p$, there exists an $n \in \mathbb{N}$ such that $\mathsf{Conv}(\{u, v\}) \subseteq L_{n,p}$.*

Proof. Let k and l be such that $u \in L_{k,p}$ and $v \in L_{l,p}$. Take $n = kl$. The result follows from Lemma 24. ■

We proceed with the proofs of Lemma 31 and Theorem 30.

Proof of Lemma 31. If p is not a time-elapsing progress cycle, then L_p is a singleton that contains the valuation in which all clocks are equal to zero. In this case, the result is immediate.

Assume that p contains a time-elapsing region. For $u, v \in L_p$, let $n \in \mathbb{N}$ be given by Lemma 37. We are in the conditions of Lemma 32: for all $x \in \mathsf{Conv}(\{u, v\})$ there exists a limit cycle π on x with $\mathsf{Duration}(\pi) > 0$ and $|\pi| \leqslant nW$ where W is the number of regions of $\mathcal{A}$. Therefore, there exists a limit cycle π' on x with $\mathsf{Duration}(\pi') \geqslant 1/2$ and $|\pi'| \leqslant 2nW$. Let $N = 2nW$ and take $\delta = \frac{1}{2}\varepsilon K_\varepsilon^N$. By Lemma 36, for all $y \in \mathcal{N}_\infty(x, \delta) \cap [p_0]$ there exists a trajectory π in $[\![\mathcal{A}]\!]_0^\varepsilon$ such that $\mathsf{first}(\pi) = x$ and $\mathsf{last}(\pi) = y$. Finally, the result follows from the fact that $L_p \subseteq [p_0]$. ■

Proof of Theorem 30. For $u, v \in L_p$, let δ as given by Lemma 31 and let $k = \lceil \frac{1}{\delta} \rceil$. Consider the points $x_0 = u$, $x_k = v$, and $x_i = u + i\delta(v - u)$ for $i = 1, \dots, k-1$. It is easy to see that $d_\infty(x_i, x_{i+1}) \leqslant \delta \cdot d_\infty(u, v) \leqslant \delta$ (because the ∞-distance between two points of a region is at most 1). Thus from Lemma 31, for all $0 \leqslant i \leqslant k-1$ there exists a trajectory from x_i to x_{i+1} in $[\![\mathcal{A}]\!]_0^\varepsilon$, and thus a trajectory π such that $\mathsf{first}(\pi) = u$ and $\mathsf{last}(\pi) = v$. ■

Soundness of Algorithm 1.

The second part of the next theorem corresponds to [Pur98, Theorem 7.3].

Theorem 38 *Let $\mathcal{A}$ be a timed automaton. Let $p = p_0\, p_1 \, \dots \, p_N$ be a progress cycle of the region graph of $\mathcal{A}$. For all $x, y \in [p_0]$, we have:*

- *For all $\Delta \in \mathbb{R}_{>0}$, there exists a trajectory π in $[\![\mathcal{A}]\!]_\Delta^0$ such that $\mathsf{first}(\pi) = x$ and $\mathsf{last}(\pi) = y$;*
- *For all $\varepsilon \in \mathbb{R}_{>0}$, there exists a trajectory π' in $[\![\mathcal{A}]\!]_0^\varepsilon$ such that $\mathsf{first}(\pi') = x$ and $\mathsf{last}(\pi') = y$.*

Proof. From Theorem 23, there exist $u, v \in L_p$ and two trajectories π_1 and π_3 of $[\![\mathcal{A}]\!]$ such that $\mathsf{first}(\pi_1) = x$ and $\mathsf{last}(\pi_1) = u$, and $\mathsf{first}(\pi_3) = v$ and $\mathsf{last}(\pi_3) = y$. By Theorem 28, there exists a trajectory π_2 of $[\![\mathcal{A}]\!]_\Delta^0$ such that $\mathsf{first}(\pi_2) = u$ and $\mathsf{last}(\pi_2) = v$. We construct π by concatenating the three trajectories π_1, π_2 and π_3. The proof is similar for the second part of the theorem, based on Theorem 30. ■

As a consequence, we get the following extension of [Pur98, Theorem 5.1].

Theorem 39 *Let $\mathcal{A}$ be a timed automaton* satisfying Assumption 9. *Let J^* be the set computed by Algorithm 1. Then $J^* \subseteq R^0_{\Delta\to 0}$ and $J^* \subseteq R_0^{\varepsilon\to 0}$.*

Proof. For all $\Delta > 0$, if a set of regions J^* is reachable in $[\![\mathcal{A}]\!]_\Delta^0$, then:

- so is the set $\mathsf{Reach}(G, J^*)$ of regions reachable from J^* in the region graph G of $\mathcal{A}$;
- any cycle p is a progress cycle, so that Theorem 38 applies: if p_0 is a region in p such that $[p_0] \cap J^* \neq \varnothing$, then the set $J^* \cup [p_0]$ is reachable in $[\![\mathcal{A}]\!]_\Delta^0$.

Since J^* is obtained by iterating the above two operations (lines 3, 5 and 6 of the algorithm) from the set of initial states $[q_0]$, this ensures that $J^* \subseteq \mathsf{Reach}([\![\mathcal{A}]\!]_\Delta^0)$. This holds for all $\Delta > 0$, and hence $J^* \subseteq R^0_{\Delta\to 0}$.

The proof for drifts on clocks is similar. ■

6.3 Completeness of Algorithm 1: $R^{\varepsilon\to 0}_{\Delta\to 0} \subseteq J^*$

To prove the completeness of Algorithm 1, we have to show that any state that is reachable in the semantics $[\![\mathcal{A}]\!]_\Delta^\varepsilon$ no matter how small are ε and Δ, lies in the set J^* computed by Algorithm 1. First, we show that if the number of transitions in trajectories is *fixed*, then there is a bound on the distance between a state reachable in $[\![\mathcal{A}]\!]_\Delta^\varepsilon$ and the set of reachable states $\mathsf{Reach}([\![\mathcal{A}]\!])$ in the classical semantics (Theorem 44). This bound vanishes when $\varepsilon \to 0$ and $\Delta \to 0$. This shows that a state $x \in R^{\varepsilon\to 0}_{\Delta\to 0}$ that is reachable in a fixed number of steps from the initial states in $[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}$ for all $\varepsilon, \Delta > 0$ is at distance zero from the reachable states in the classical semantics. An argument related to topologically closed sets then shows that $x \in \mathsf{Reach}([\![\mathcal{A}]\!])$. Second, to extend the result to the whole set $R^{\varepsilon\to 0}_{\Delta\to 0}$, we roughly use the fact that longer trajectories necessarily contain a cycle that is added to J^* by the algorithm. Therefore, only a bounded part of those trajectories can take the state far from J^*. By a similar argument as above, this distance to J^* is shown to vanish when $\varepsilon \to 0$ and $\Delta \to 0$ (Theorem 45). The proofs of the theorems are based on a detailed study of the reachability properties of the perturbed semantics, for which we need parametric DBM, an extension of DBM (that we have presented in Section 4).

A *parametric* DBM (PDBM) in $\mathbb{R}^n$ is a matrix $\mathbf{M} = \big(m_{i,j}\big)_{0\leqslant i,j\leqslant n}$ where $m_{i,j} \in \mathbb{Z}\times\mathbb{N}$ is called a *parametric bound*. In a PDBM, each $m_{i,j}$ is a couple (a, b) of integers with $b \geqslant 0$. Given a number $\Omega \in \mathbb{R}_{\geqslant 0}$, the *value* of $m_{i,j}$ is $[\![m]\!]_{\Omega} = a + b\Omega$. The set represented by $\mathbf{M}$ is:

$$[\![\mathbf{M}]\!]_{\Omega} = \{(x_1, \dots, x_n) \in \mathbb{R}^n \mid \forall\, 0 \leqslant i, j \leqslant n : \ x_i - x_j \leqslant [\![m_{ij}]\!]_{\Omega} \wedge x_0 = 0\}.$$

As usual, we often write $[\![\mathbf{M}]\!]$ for $[\![\mathbf{M}]\!]_0$. More general definitions of PDBM have been introduced in [AAB00, HRSV01], with implementations. Here, we use PDBM for purely theoretical purposes, so we keep the definition as simple as possible.

For a PDBM $\mathbf{M} = \big(m_{ij}\big)_{0\leqslant i,j\leqslant n}$ with $m_{ij} = (a_{ij}, b_{ij})$, we define the *width* of $\mathbf{M}$ by $w(\mathbf{M}) = \max\{b_{ij} \mid 0 \leqslant i, j \leqslant n\}$. Thus a DBM is a zero-width PDBM. Any closed rectangular guard g can be represented by a PDBM $\mathbf{M}_g$ with $w(\mathbf{M}_g) = 2$ such that for all $\Omega \in \mathbb{R}_{\geqslant 0}$ we have $[\![\mathbf{M}_g]\!]_{\Omega} = \mathcal{N}_{\infty}([\![g]\!], \Omega)$. In particular, $[\![g]\!] = [\![\mathbf{M}_g]\!]$.

Example Let $g \equiv x = 4 \wedge 1 \leqslant y \leqslant 3$. Then,

$$\mathbf{M}_g = \begin{array}{c} \\ \mathsf{0} \\ \mathsf{x} \\ \mathsf{y} \end{array}\!\!\begin{array}{c} \begin{array}{ccc} \mathsf{0} & \mathsf{x} & \mathsf{y} \end{array} \\ \begin{pmatrix} (0,0) & (-4,1) & (-1,1) \\ (4,1) & (0,0) & (3,2) \\ (3,1) & (-1,2) & (0,0) \end{pmatrix} \end{array}$$

When the reachable states in the perturbed semantics of a timed automaton $\mathcal{A}$ are computed parametrically using PDBM, it would be nice that the classical semantics $[\![\mathbf{M}]\!]$ gives exactly the reachable states in $[\![\mathcal{A}]\!]$ and that the perturbed semantics $[\![\mathbf{M}]\!]_{\Omega}$ gives the reachable states in $[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}$. This can be obtained when $\varepsilon = 0$ by taking $\Omega = \Delta$. For the general case $\varepsilon > 0$, the set $[\![\mathbf{M}]\!]_{\Omega}$ over-approximates the reachable states, provided ε is sufficiently small. We are more precise in Lemma 42 and Lemma 43.

In that context, the width of PDBM records the accumulation of the deviations allowed by the perturbed semantics. This is useful to bound the distance between states that are reachable in the perturbed semantics and states that are reachable in the classical semantics. The following lemma gives such a bound.

Lemma 40 (See also [Pur98, Lemma 7.4**])** *Let* $\mathbf{M}$ *be a PDBM in* $\mathbb{R}^n$ *and let* $\Omega \in \mathbb{R}_{\geqslant 0}$ *such that* $\Omega \cdot (2n+1) \cdot w(\mathbf{M}) < 1$. *Let* $Z = [\![\mathbf{M}]\!]$ *and* $Z' = [\![\mathbf{M}]\!]_{\Omega}$. *For all* $x' \in Z'$, *there exists* $x \in Z$ *such that* $\|x' - x\|_{\infty} \leqslant n \cdot w(\mathbf{M}) \cdot \Omega$.

Proof. First, assume that x' is a vertex of Z'. Then x' can be obtained by solving a system of n equations of the form $x'_i - x'_j = [\![m_{ij}]\!]_{\Omega}$, $x'_i = [\![m_{i0}]\!]_{\Omega}$ or $x'_i = -[\![m_{0i}]\!]_{\Omega}$. Therefore, each x'_i is the sum or difference of at most n coefficients $[\![m_{ij}]\!]_{\Omega}$. Since the bounds m_{ij} are entries of $\mathbf{M}$, for all $1 \leqslant i \leqslant n$, if $x'_i = l_i + k_i\Omega$ for some $l_i, k_i \in \mathbb{Z}$, then $|k_i| \leqslant n \cdot w(\mathbf{M})$ and we take $x_i = l_i$. Then $\|x' - x\|_\infty \leqslant n \cdot w(\mathbf{M}) \cdot \Omega$ and we claim that $x \in Z$. Let $l_0 = k_0 = 0$. Then, for all $0 \leqslant i, j \leqslant n$ we have (for $m_{ij} = (a_{ij}, b_{ij})$):

$$x'_i - x'_j = l_i - l_j + (k_i - k_j) \cdot \Omega \leqslant a_{ij} + b_{ij}\Omega$$

Hence,

$$l_i - l_j \leqslant a_{ij} + (b_{ij} - k_i + k_j) \cdot \Omega$$

Since l_i, l_j and a_{ij} are integers and $|(b_{ij} - k_i + k_j) \cdot \Omega| \leqslant (2n+1) \cdot w(\mathbf{M}) \cdot \Omega < 1$, we have $x_i - x_j = l_i - l_j \leqslant a_{ij} = [\![m_{ij}]\!]_0$. Therefore $x \in Z$.

Second, if x' is not a vertex, then it can be written as $x' = \sum_i \lambda_i v'_i$ with $\lambda_i \geqslant 0$ and $\sum_i \lambda_i = 1$ and each v'_i is a vertex of Z'. From the proof above, for each v'_i there exists $v_i \in Z$ such that $\|v'_i - v_i\|_\infty \leqslant n \cdot w(\mathbf{M}) \cdot \Omega$. We take $x = \sum_i \lambda_i v_i$. Clearly $x \in Z$, and we have:

$$\begin{aligned}\|x' - x\|_\infty = \Big\| \sum_i \lambda_i (v'_i - v_i) \Big\|_\infty &\leqslant \sum_i \lambda_i \|v'_i - v_i\|_\infty \\ &\leqslant \sum_i \lambda_i (n \cdot w(\mathbf{M}) \cdot \Omega) \leqslant n \cdot w(\mathbf{M}) \cdot \Omega\end{aligned}$$

■

Now, we show how to extend to PDBM the operations that we have presented in Section 4 for DBM. To do so, we have to define the minimum of two parametric bounds (for intersection of PDBM). We define a lexicographic order on parametric bounds: $(a, b) \leqslant (a', b')$ if and only of either $a < a'$, or $a = a'$ and $b \leqslant b'$. This (syntactical) definition is justified by the following observation: for all Ω such that $b\Omega \leqslant 1$, if $(a, b) \leqslant (a', b')$ then $[\![(a, b)]\!]_\Omega \leqslant [\![(a', b')]\!]_\Omega$. Thus if we take a sufficiently small Ω, the order is preserved at the semantic level. In the sequel, this will imply that provided Ω is below some threshold, the operations on PDBM can be performed *independently* of the value of Ω. The *sum* of two parametric bounds (a, b) and (a', b') is $(a + a', b + b')$.

We review the important operations on PDBM:

- Intersection: the intersection of two PDBM $\mathbf{M}_1$ and $\mathbf{M}_2$ is the PDBM $\mathbf{M}$ whose entries are the minimum (according to the lexicographic order on parametric bounds) of the corresponding entries of $\mathbf{M}_1$ and $\mathbf{M}_2$. Hence $w(\mathbf{M}) \leqslant \max\{w(\mathbf{M}_1), w(\mathbf{M}_2)\}$.
- Time passing and reset: those operations only substitute entries of the matrix with other entries of the matrix and they preserve the normal form. Thus the width cannot increase.
- Normalization: to obtain the normal form of a PDBM $\mathbf{M}$ in $\mathbb{R}^n$, each entry m_{ij} is replaced by the length of the shortest path from node i to node j, which has at most n edges. Therefore, the width of the normal form is bounded by $n \cdot w(\mathbf{M})$.
- Emptiness test: given a PDBM $\mathbf{M}$, let $\mathbf{M}'$ be its normal form. The emptiness test checks whether one of the diagonal entries is negative (a parametric bound $m = (a, b)$ is *negative* iff $m < (0, 0)$ iff $a \leqslant -1$).

A summary of the above observations is given in Table 1. Their correctness is established in the following lemma.

Lemma 41 *For all PDBM* $\mathbf{M}$, $\mathbf{M}'$ *in* $\mathbb{R}^n$, *we have:*

PDBM in $\mathbb{R}^n$	Input in NF	Output in NF	Width of the result
Intersection $\mathbf{M_1} \cap \mathbf{M_2}$	No	No	$\leqslant \max\{w(\mathbf{M_1}), w(\mathbf{M_2})\}$
Time passing $\mathbf{M}\nearrow$	Yes	Yes	$\leqslant w(\mathbf{M})$
Reset $\mathbf{M}[R := 0]$	Yes	Yes	$\leqslant w(\mathbf{M})$
Normalization of $\mathbf{M}$ Emptiness test of $\mathbf{M}$	No	Yes	$\leqslant n \cdot w(\mathbf{M})$

Table 1 Operations on PDBM (NF = normal form).

- $[\![\mathbf{M} \cap \mathbf{M}']\!]_\Omega = [\![\mathbf{M}]\!]_\Omega \cap [\![\mathbf{M}']\!]_\Omega$ *for all* $0 \leqslant \Omega \leqslant 1/\max\{w(\mathbf{M}), w(\mathbf{M}')\}$*;*
- $[\![\mathbf{M}\nearrow]\!]_\Omega = [\![\mathbf{M}]\!]_\Omega\nearrow$ *for all* $\Omega \in \mathbb{R}_{\geqslant 0}$*, if* $\mathbf{M}$ *is in normal form;*
- $[\![\mathbf{M}[R := 0]]\!]_\Omega = [\![\mathbf{M}]\!]_\Omega[R := 0]$ *for all* $\Omega \in \mathbb{R}_{\geqslant 0}$ *and all* $R \subseteq \{x_1, \dots, x_n\}$*, if* $\mathbf{M}$ *is in normal form;*
- *if* $\mathbf{M}'$ *is the normal form of* $\mathbf{M}$*, then the DBM* $\left([\![m'_{ij}]\!]_\Omega\right)_{0 \leqslant i,j \leqslant n}$ *is the normal form of the DBM* $\left([\![m_{ij}]\!]_\Omega\right)_{0 \leqslant i,j \leqslant n}$*, for all* $0 \leqslant \Omega \leqslant 1/\max\{w(\mathbf{M}), w(\mathbf{M}')\}$*;*
- $[\![\mathbf{M}]\!]_\Omega = \varnothing$ *iff* $[\![\mathbf{M}]\!]_0 = \varnothing$*, for all* $0 \leqslant \Omega \leqslant 1/(n \cdot w(\mathbf{M}))$*.*

Proof. The argument is similar for the five claims. We give the details for the last one. First, we have $[\![\mathbf{M}]\!]_0 \subseteq [\![\mathbf{M}]\!]_\Omega$ for all Ω. Thus it suffices to show that $[\![\mathbf{M}]\!]_0 = \varnothing$ implies that $[\![\mathbf{M}]\!]_\Omega = \varnothing$ for all $\Omega < 1/(n \cdot w(\mathbf{M}))$. If $[\![\mathbf{M}]\!]_0 = \varnothing$ then there exists a parametric bound $m' = (a, b)$ in the diagonal of the normal form PDBM $\mathbf{M}'$ such that $a \leqslant -1$. Since $b \leqslant n \cdot w(\mathbf{M})$, we have $[\![m']\!]_\Omega = a + b\Omega < 0$ and therefore $[\![\mathbf{M}]\!]_\Omega$ is empty. ■

Notations. Given a TTS $\mathcal{T} = \langle S, \iota, \Sigma, \rightarrow \rangle$, let $U \subseteq S$ and $\sigma \in \Sigma$. We define the following operators:

$$\mathsf{post}^{\sigma}_{\mathcal{T}}(U) = \{s' \in S \mid \exists s \in U.\ s \xrightarrow{\sigma} s'\}$$
$$\mathsf{post}^{\mathsf{time}}_{\mathcal{T}}(U) = \{s' \in S \mid \exists s \in U.\ \exists t \in \mathbb{R}_{\geqslant 0}.\ s \xrightarrow{t} s'\}$$

We use the PDBM to characterize the relationship between the reachable states of the classical semantics $[\![\mathcal{A}]\!]$ and those of the perturbed semantics $[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}$.

By an abuse of notation, we omit the location in the argument of $\mathsf{post}(\cdot)$, that is we use $Z = [\![\mathbf{M}]\!]$ instead of $Z = \{\ell\} \times [\![\mathbf{M}]\!]$ for $\ell \in \mathsf{Loc}$. Finally, we assume that the edges of timed automata are identified by their label. This is clearly not restrictive for reachability analysis.

In Lemma 42, the PDBM $\mathbf{M}'$ contains the exact information about the timed successors of $\mathbf{M}$ in the classical semantics, and it is an over-approximation of the timed successors in the perturbed semantics. Lemma 43 is similar for discrete successors.

Lemma 42 *Let* $\mathcal{A}$ *be a timed automaton with* n *clocks and largest constant* M*. Let* $\mathbf{M}$ *be a PDBM in* $\mathbb{R}^n$ *in normal form. There exists a PDBM* $\mathbf{M}'$ *in normal form such that:*

- $\forall \Omega \in \mathbb{R}_{\geqslant 0} \cdot \forall \Delta \in \mathbb{R}_{\geqslant 0} \cdot \forall \varepsilon \leqslant \Omega/(2(M+1)) : \mathsf{post}^{\mathsf{time}}_{[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}}([\![\mathbf{M}]\!]_\Omega) \subseteq [\![\mathbf{M}']\!]_\Omega$*;*
- $\mathsf{post}^{\mathsf{time}}_{[\![\mathcal{A}]\!]^{0}_{0}}([\![\mathbf{M}]\!]_0) = [\![\mathbf{M}']\!]_0$*;*
- $w(\mathbf{M}') = w(\mathbf{M}) + 1$.

Proof. Assume that $\Omega, \Delta \in \mathbb{R}_{\geqslant 0}$ and $\varepsilon \leqslant \Omega/(2(M+1))$. First, observe that in the classical semantics $[\![\mathcal{A}]\!]$, the length of a timed transition is bounded by M. In the perturbed semantics $[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}$ however, a timed transition may be longer than M because clocks can progress more slowly, namely at the rate $1-\varepsilon$. Therefore, the length of a timed transition is bounded by $M/(1-\varepsilon)$ and thus by $M+1$ since $\varepsilon \leqslant 1/(M+1)$. Second, we obtain $\mathbf{M}'$ by constructing the time successor of $\mathbf{M}$ as described above (in the exact semantics), and then by replacing each bound (a,b) of the PDBM by $(a,b+1)$, except on the diagonal. Clearly we have $w(\mathbf{M}') = w(\mathbf{M})+1$ and $[\![\mathbf{M}\nearrow]\!]_0 = [\![\mathbf{M}']\!]_0$ and thus $\mathsf{post}^{\mathsf{time}}_{[\![\mathcal{A}]\!]^0_0}([\![\mathbf{M}]\!]_0) = [\![\mathbf{M}']\!]_0$. On the other hand, if we have $(\ell, x) \xrightarrow{t} (\ell, x')$ in $[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}$ and $x_i - x_j \leqslant [\![m_{ij}]\!]_{\Omega}$, then:

$$x'_i - x'_j \leqslant [\![m_{ij}]\!]_{\Omega} + 2\varepsilon t \leqslant [\![m_{ij}]\!]_{\Omega} + 2\varepsilon(M+1) \leqslant [\![m_{ij}]\!]_{\Omega} + \Omega = [\![m'_{ij}]\!]_{\Omega}$$

Therefore $\mathsf{post}^{\mathsf{time}}_{[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}}([\![\mathbf{M}]\!]_{\Omega}) \subseteq [\![\mathbf{M}']\!]_{\Omega}$. ∎

Lemma 43 *Let $\mathcal{A}$ be a timed automaton with n clocks and alphabet Lab. Let $\mathbf{M}$ be a PDBM in $\mathbb{R}^n$. For all $\sigma \in \mathsf{Lab}$, there exists a PDBM $\mathbf{M}'$ in normal form such that:*

- $\forall \Omega \leqslant 1/(\max\{2, w(\mathbf{M})\}) \cdot \forall \Delta \leqslant \Omega \cdot \forall \varepsilon \in \mathbb{R}_{\geqslant 0} : \mathsf{post}^{\sigma}_{[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}}([\![\mathbf{M}]\!]_{\Omega}) \subseteq [\![\mathbf{M}']\!]_{\Omega}$;
- $\mathsf{post}^{\sigma}_{[\![\mathcal{A}]\!]^0_0}([\![\mathbf{M}]\!]_0) = [\![\mathbf{M}']\!]_0$;
- $w(\mathbf{M}') \leqslant n \cdot \max\{2, w(\mathbf{M})\}$.

Proof. Assume that $\Omega, \varepsilon \in \mathbb{R}_{\geqslant 0}$ and $\Delta \leqslant \Omega$. Let $(\ell, \ell', g, \sigma, R)$ be the edge of $\mathcal{A}$ associated with σ. Let $\mathbf{M}_g$ be the PDBM that represents the guard g (with $w(\mathbf{M}_g) = 2$). To construct $\mathbf{M}'$, let $\mathbf{M}_{\cap}$ be the PDBM $\mathbf{M} \cap \mathbf{M}_g$ put in normal form, and let $\mathbf{M}' = \mathbf{M}_{\cap}[R := 0]$ which is in normal form. According to Table 1, we have $w(\mathbf{M}') \leqslant n \cdot \max\{2, w(\mathbf{M})\}$ and

$$\mathsf{post}^{\sigma}_{[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}}([\![\mathbf{M}]\!]_{\Omega}) \subseteq \mathsf{post}^{\sigma}_{[\![\mathcal{A}]\!]^{\varepsilon}_{\Omega}}([\![\mathbf{M}]\!]_{\Omega}) = [\![\mathbf{M}']\!]_{\Omega} \quad \text{(by Lemma 41)}$$

For $\Omega = \Delta = \varepsilon = 0$, the sets collapse and $\mathsf{post}^{\sigma}_{[\![\mathcal{A}]\!]^0_0}([\![\mathbf{M}]\!]_0) = [\![\mathbf{M}']\!]_0$. ∎

With the previous two lemmas, we have characterized how much the set of reachable states can increase by taking *one* transition (either timed or discrete) in the perturbed semantics $[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}$ instead of the classical semantics $[\![\mathcal{A}]\!]^0_0$. That increase is measured in terms of the width of a PDBM. In the next theorem, we use an argument by induction to give a bound on the increase after a given number of transitions. However, this is not sufficient to prove the completeness of Algorithm 1. We need in addition to show that every trajectory π' in $[\![\mathcal{A}]\!]^{\varepsilon}_{\Delta}$ can be approached by a trajectory π in $[\![\mathcal{A}]\!]^0_0$ where each intermediate state in π is "close" to the corresponding state in π'. To obtain this result, we introduce the notion of *automaton refinement* that roughly divides the guards into small pieces of size[6] $1/\gamma$ (with $\gamma \in \mathbb{N}$) so that two valuations that satisfy the same guard are necessarily "close" to each other (by choosing γ sufficiently large). This is the core of Theorem 44.

[6] The *size* of a set is the maximal distance (for d_{∞}) between two points in the set.

Automaton refinement. Given a timed automaton $\mathcal{A}$ with n clocks and a positive integer γ, the *γ-refinement* of $\mathcal{A}$ is the timed automaton $\mathcal{A}_\gamma$ constructed from $\mathcal{A}$ as follows:

– we first substitute each constant c appearing in the rectangular constraints (guards, invariants, initial and final conditions) of $\mathcal{A}$ with $c\gamma$
– we then replace each edge (l, l', g, σ, R) in the resulting automaton with the set of all edges (l, l', g', σ, R) where g' ranges over the set of all *unit constraints* (*i.e.*, conjunctions of constraints of the form $x_i = a$ or $a \leqslant x_i \leqslant a + 1$) that imply g. Equalities of the form $x_i = a$ are used only if g implies $x_i = a$.

Roughly, the γ-refinement of $\mathcal{A}$ is a scaling of the constants by a factor γ (and thus a scaling of the time), followed by a partitioning of the guards such that the distance between two valuations that satisfy the guard is at most 1 (instead of being a multiple of γ, as is the case after the first step).

The important property of such refinements is that for all $\Delta, \varepsilon \in \mathbb{R}_{\geqslant 0}$, the two TTS $[\![\mathcal{A}]\!]^\varepsilon_\Delta$ and $[\![\mathcal{A}_\gamma]\!]^\varepsilon_{\gamma\Delta}$ are bisimilar, witnessed by the bijection $\mu_\gamma : Q_\mathcal{A} \to Q_{\mathcal{A}_\gamma}$ such that $\mu_\gamma(\ell, v) = (\ell, \gamma v)$. We extend μ_γ to trajectories as expected (states are mapped according to μ_γ and the time stamps are multiplied by γ). Finally, for all $v, v' \in Q_{\mathcal{A}_\gamma}$ we have $\|\mu_\gamma^{-1}(v) - \mu_\gamma^{-1}(v')\|_\infty = \|v - v'\|_\infty/\gamma$

Example. For $\gamma = 2$, an edge $(\ell, \ell', g, \sigma, R)$ in $\mathcal{A}$ with $g \equiv (x = 4) \wedge (1 \leqslant y \leqslant 3)$ is replaced in $\mathcal{A}_\gamma$ with the following four edges:

$$(\ell, \ell', \{x = 8 \wedge 2 \leqslant y \leqslant 3\}, \sigma, R) \quad (\ell, \ell', \{x = 8 \wedge 4 \leqslant y \leqslant 5\}, \sigma, R)$$
$$(\ell, \ell', \{x = 8 \wedge 3 \leqslant y \leqslant 4\}, \sigma, R) \quad (\ell, \ell', \{x = 8 \wedge 5 \leqslant y \leqslant 6\}, \sigma, R)$$

The next theorem extends and clarifies Theorem 8.2 in [Pur98].

Theorem 44 *Let $\mathcal{A}$ be a timed automaton with $n \geqslant 1$ clocks and largest constant M. For all distances $0 < \alpha < 1$, for any number of steps $k \in \mathbb{N}$, there exist two numbers $D, E \in \mathbb{R}_{>0}$ such that for all $\Delta \in [0, D]$, for all $\varepsilon \in [0, E]$ and for all stutter-free trajectories π' of $[\![\mathcal{A}]\!]^\varepsilon_\Delta$ such that $|\pi'| = k$, there exists a trajectory π of $[\![\mathcal{A}]\!]$ such that:*

– $\mathsf{first}(\pi) \in [\mathsf{first}(\pi')]$;
– $\mathsf{trace}(\pi) = \mathsf{trace}(\pi')$;
– *π is "close" to π' in the following sense: $\forall\, 0 \leqslant i \leqslant k$, if $\mathsf{state}_i(\pi) = (\ell_i, v_i)$ and $\mathsf{state}_i(\pi') = (\ell'_i, v'_i)$, then $\ell_i = \ell'_i$ and $\|v_i - v'_i\|_\infty < \alpha$.*

Proof. Given $0 < \alpha < 1$ and $k \in \mathbb{N}$, let $\gamma = \lceil 2/\alpha \rceil$ and:

$$D = \frac{\alpha}{4\gamma(n+1)^{k+1}} \qquad\qquad E = \frac{D}{2(\gamma M + 1)}$$

Let $\Delta \in [0, D]$ and $\varepsilon \in [0, E]$ and let $\Omega = \gamma D$. Let $\mathsf{trace}(\pi') = \sigma_1\sigma_2\ldots\sigma_k$. Let $\rho' = \mu_\gamma(\pi')$. Then ρ' is a trajectory of $[\![\mathcal{A}_\gamma]\!]^\varepsilon_{\gamma\Delta}$. Let $\mathbf{M}_0$ be a PDBM in normal form such that $[\![\mathbf{M}_0]\!] = [\mathsf{first}(\pi')]$ and $w(\mathbf{M}_0) = 0$ (in fact $\mathbf{M}_0$ can be seen as a DBM). Observe that $\gamma\Delta \leqslant \Omega$ and $\varepsilon \leqslant \Omega/(2(M_\gamma + 1))$ where $M_\gamma = \gamma M$ is the largest constant of $\mathcal{A}_\gamma$. Therefore, by Lemma 42 and 43, there exists PDBM $\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_k$ in normal form such that, for all $1 \leqslant i \leqslant k$ and provided that $\Omega \leqslant 1/(\max\{2, w(\mathbf{M}_i)\})$,

(a) $\mathsf{post}^{\sigma_i}_{[\![\mathcal{A}_\gamma]\!]^\varepsilon_{\gamma\Delta}}([\![\mathbf{M}_{i-1}]\!]_\Omega) \subseteq [\![\mathbf{M}_i]\!]_\Omega$;

(b) $\mathsf{post}^{\sigma_i}_{[\![\mathcal{A}_\gamma]\!]}([\![\mathbf{M}_{i-1}]\!]_0) = [\![\mathbf{M}_i]\!]_0$;
(c) $w(\mathbf{M}_i) \leqslant \max\{w(\mathbf{M}_{i-1}) + 1, n \cdot \max\{2, w(\mathbf{M}_{i-1})\}$.

Let us show that $w(\mathbf{M}_i) \leqslant 2(n+1)^i$. We proceed by induction. The claim holds for $i = 1$ since $w(\mathbf{M}_0) = 0$. Assume that it holds up to some index $i-1$, with $i \geqslant 2$. Then we have

$$\begin{aligned} w(\mathbf{M}_i) &\leqslant \max\left\{w(\mathbf{M}_{i-1}) + 1,\ n \cdot \max\{2, w(\mathbf{M}_{i-1})\}\right\} \\ &\leqslant \max\{1 + 2(n+1)^{i-1},\ 2n \cdot (n+1)^{i-1}\} && \text{(by (c))} \\ &\leqslant 2(n+1)^{i-1} + 2n \cdot (n+1)^{i-1} \\ &\leqslant 2(n+1)^i \end{aligned}$$

Notice that the condition $\Omega \leqslant 1/(\max\{2, w(\mathbf{M}_i)\})$ is satisfied. For each $0 \leqslant i \leqslant k$, let $q'_i = \mathsf{state}_i(\pi')$. By (a), we have $\mu_\gamma(q'_k) \in [\![\mathbf{M}_k]\!]_\Omega$. Since $\alpha < 1$, it is easy to see that:

$$\Omega = \frac{\alpha}{4(n+1)^{k+1}} < \frac{1}{(2n+1)w(\mathbf{M}_k)}$$

and thus by Lemma 40, there exists $q_k \in [\![\mathbf{M}_k]\!]$ such that:

$$\|\mu_\gamma(q'_k) - q_k\|_\infty \leqslant n \cdot w(\mathbf{M}_k) \cdot \Omega < 2(n+1)^{k+1} \cdot \Omega \leqslant \alpha.$$

Using (b), we can construct in a backward fashion a trajectory ρ of $[\![\mathcal{A}_\gamma]\!]$ such that:

– $\mathsf{last}(\rho) = q_k$;
– $\mathsf{trace}(\rho) = \mathsf{trace}(\pi')$;
– $\mathsf{first}(\rho) \in [\![\mathbf{M}_0]\!] = [q'_0]$.

For each $0 \leqslant i \leqslant k$, let $q_i = \mathsf{state}_i(\rho)$. For all i such that $\sigma_i \neq \mathsf{time}$, we have $q_i \in [\![g'_i]\!]$ and $\mu_\gamma(q'_i) \in \mathcal{N}_\infty([\![g'_i]\!], \gamma\Delta)$ where g'_i is the guard of the edge of $\mathcal{A}_\gamma$ associated to σ_i that has been taken in ρ. Since the size of g'_i is at most 1, we have:

$$\|\mu_\gamma(q'_i) - q_i\|_\infty \leqslant 1 + \gamma\Delta \leqslant 1 + \Omega \tag{10}$$

Observe that the effect of discrete transitions is to reset some clocks and that does not increase the ∞-distance between two states: we also have $\|\mu_\gamma(q'_i) - q_i\|_\infty \leqslant 1 + \gamma\Delta$ for all i such that $\sigma_{i-1} \neq \mathsf{time}$. Since π' is stutter-free and $\mathsf{trace}(\rho) = \mathsf{trace}(\pi')$, Eq. (10) holds for all $0 \leqslant i \leqslant k$. Now, let $\pi = \mu_\gamma^{-1}(\rho)$ which is a trajectory of $[\![\mathcal{A}]\!]$ since ρ is a trajectory of $[\![\mathcal{A}_\gamma]\!]$. Thus, we have for all $0 \leqslant i \leqslant k$:

$$\|q'_i - \mu_\gamma^{-1}(q_i)\|_\infty \leqslant \frac{1+\Omega}{\gamma} < \frac{2}{\gamma} \leqslant \alpha$$

which entails that π is “close” to π' as required. ■

The following theorem is the key of the proof of completeness. It shows that for all distances $\alpha > 0$, we can choose sufficiently small values of Δ and ε such that from J^* the points that are reachable in $[\![\mathcal{A}]\!]^\varepsilon_\Delta$ are at distance at most α from J^*. By contrast with Theorem 44, we do not make the hypothesis that the length of the trajectories is bounded. This result is similar to Theorem 8.3 in [Pur98], but the constants are different because only drifting clocks were considered by Puri and the bound of Lemma 16 was wrong.

Theorem 45 *Let $\mathcal{A}$ be a timed automaton with $n \geqslant 1$ clocks and largest constant M that satisfies Assumption 9. For all distances $\alpha \in \mathbb{R}_{>0}$, there exist two numbers $D, E \in \mathbb{R}_{>0}$ such that for all $\Delta \in [0, D]$, for all $\varepsilon \in [0, E]$ and for all trajectories π' of $[\![\mathcal{A}]\!]_\Delta^\varepsilon$ such that $\mathsf{first}(\pi') \in J^*$, we have $d_\infty(\mathsf{last}(\pi'), J^*) < \alpha$.*

Proof. Without loss of generality, we may assume that $\alpha < \frac{1}{2n}$. Let W be the number of regions of $\mathcal{A}$, let $\gamma = \lceil 2/\alpha \rceil$ and:

$$D = \frac{\alpha}{4\gamma(n+1)^{2W+1}} \qquad\qquad E = \frac{D}{2(\gamma M + 1)}$$

Let $\Delta \in [0, D]$ and $\varepsilon \in [0, E]$ and let π' be a stutter-free trajectory of $[\![\mathcal{A}]\!]_\Delta^\varepsilon$ such that $\mathsf{first}(\pi') \in J^*$. Let $m = |\pi'|$ and for each $0 \leqslant i \leqslant m$, let $q'_i = state_i(\pi')$.

- If $m \leqslant 2W$. By Theorem 44, there exists a trajectory π of $[\![\mathcal{A}]\!]$ such that $\mathsf{first}(\pi) \in [\mathsf{first}(\pi')]$ and for all $0 \leqslant i \leqslant m$, $\|q_i - q'_i\|_\infty < \alpha$ where $q_i = \mathsf{state}_i(\pi)$. Since $q_0 \in [q'_0] \subseteq J^*$, the state q_m is reachable from J^* and thus $q_m \in J^*$. Since $\|q_m - q'_m\|_\infty < \alpha$ this yields $d_\infty(\mathsf{last}(\pi'), J^*) < \alpha$.
- If $m > 2W$. By induction, assume that $d_\infty(q'_i, J^*) < \alpha$ for all $0 \leqslant i \leqslant m - 1$. Consider the sub-trajectory of π' from state q'_{m-2W} to q'_m, and according to Theorem 44, let π be a trajectory such that for all $m - 2W \leqslant i \leqslant m$, it holds $\|q_i - q'_i\|_\infty < \alpha$, where $q_i = state_{i-(m-2W)}(\pi)$. Then for all i with $m - 2W \leqslant i \leqslant m - 1$, we have :

  $$d_\infty(q_i, J^*) \leqslant \|q_i - q'_i\|_\infty + d_\infty(q'_i, J^*) \leqslant 2\alpha < \frac{1}{n},$$

  and by Lemma 16, this implies $[q_i] \cap J^* \neq \varnothing$ for all $m - 2W \leqslant i \leqslant m - 1$ (since J^* is a zone-set).
  On the other hand, the trajectory π has the same trace as the sub-trajectory of π' from q'_{m-2W} to q'_m and thus it is stutter-free and $|\pi| = 2W$. Therefore, π has $2W + 1$ states and thus there exists two states q_k and $q_{k'}$ in π with $k < k'$ such that $[q_k] = [q_{k'}]$ and a discrete transition occurred along π between q_k and $q_{k'}$ in π, and thus there exists a path from $[q_k]$ to itself in the region graph of $\mathcal{A}$. Since $[q_k] \cap J^* \neq \varnothing$, we have $[q_k] \subseteq J^*$ by line 5 of Algorithm 1 and $[q_i] \subseteq J^*$ for all $i \geqslant k$ by line 6 of the algorithm. So we have $q_m \in J^*$ and since $\|q_m - q'_m\|_\infty < \alpha$, this yields $d_\infty(\mathsf{last}(\pi'), J^*) < \alpha$. ■

Theorem 46 *Let J^* be the set computed by Algorithm 1. Then $R_{\Delta\to 0}^{\varepsilon\to 0} \subseteq J^*$.*

Proof. For all $y \in R_{\Delta\to 0}^{\varepsilon\to 0}$, for all $\Delta > 0$ and $\varepsilon > 0$ there exists a trajectory π of $[\![\mathcal{A}]\!]_\Delta^\varepsilon$ such that $\mathsf{first}(\pi) \in J^*$ (because J^* contains the initial states) and $\mathsf{last}(\pi) = y$. Therefore, by Theorem 45 for all $\alpha \in \mathbb{R}_{>0}$ we have $d_\infty(y, J^*) < \alpha$. This implies that $d_\infty(y, J^*) = 0$ and since J^* is a closed set (a finite union of closed regions) we have $y \in J^*$. ■

With Theorems 39 and 46 we have proven the following inclusions:

$$\begin{array}{ccccccc} & & & R^{0}_{\Delta\to 0} & & & \\ & & \rotatebox{45}{$\subseteq$} & & \rotatebox{-45}{$\subseteq$} & & \\ R^{\varepsilon\to 0}_{\Delta\to 0} & \subseteq & J^* & & & & R^{\varepsilon\to 0}_{\Delta\to 0} \\ & & \rotatebox{-45}{$\subseteq$} & & \rotatebox{45}{$\subseteq$} & & \\ & & & R^{\varepsilon\to 0}_{0} & & & \end{array}$$

All those sets are thus equal:

Theorem 47 *Under Assumption 9, we have $R^0_{\Delta\to 0} = R^{\varepsilon\to 0}_0 = R^{\varepsilon\to 0}_{\Delta\to 0}$, and those sets are computed by Algorithm 1.*

7 Complexity

The complexity issues have been studied in [Pur98]. We mention the main theorem and we give a detailed proof of the hardness result.

Theorem 48 ([Pur98]) *Given a timed automaton $\mathcal{A} = \langle \mathsf{Loc}, \mathsf{Var}, q_0, \mathsf{Lab}, \mathsf{Edg}\rangle$ satisfying Assumption 9 and a location $\ell \in \mathsf{Loc}$, deciding whether there exists a valuation v such that $(\ell, v) \in R^0_{\Delta\to 0}$ (or equivalently $(\ell, v) \in R^{\varepsilon\to 0}_0$, or $(\ell, v) \in R^{\varepsilon\to 0}_{\Delta\to 0}$) is PSPACE-complete.*

The proof uses the following definition of Linear Bounded Turing Machines (LBTM). A LBTM is a non-deterministic Turing machine that can only use a number of tape cells equal to the length of its input.

Definition 49 (Linear Bounded Turing Machine) *A* LBTM *$M = (Q, \Sigma, q_0, q_f, E)$ consists of:*

- *a finite set of control states Q,*
- *a finite alphabet Σ,*
- *an initial state $q_0 \in Q$, a final state $q_f \in Q$,*
- *and a set of transitions $E \subseteq Q \times \Sigma \times \Sigma \times \{\mathsf{left}, \mathsf{right}\} \times Q$.*

A configuration *of M is a triple $(q, w, i) \in Q \times \Sigma^* \times \mathbb{N}$ where q is a control location, $w \in \Sigma^*$ is the content of the tape, and i is the position of the tape head. A configuration (q', w', i') is a* successor *of a configuration (q, w, i) iff there exists a transition $(q, \sigma, \sigma', d, q') \in E$ such that:*

(1) *$w_i = \sigma$;*
(2) *$w'_i = \sigma'$ and $w'_j = w_j$ for all $j \neq i$;*
(3) *$i' = i - 1$ if $d = \mathsf{left}$ and $i' = i + 1$ if $d = \mathsf{right}$ with $1 \leqslant i' \leqslant |w|$.*

We assume that the condition $1 \leqslant i' \leqslant |w|$ is realized using input delimiters. An execution *of M on the input $x \in \Sigma^*$ is a sequence $s_0 s_1 \dots s_n$ of configurations starting with $s_0 = (q_0, x, 1)$ and such that s_{i+1} is a successor of s_i for every $0 \leqslant i < n$. We say that M* accepts *x iff M has an execution on x finishing in $s_n = (q_f, w, i)$ for some $w \in \Sigma^*$ and $i \in \mathbb{N}$. The* acceptance problem for LBTM *asks, given a LBTM M and an input word $x \in \Sigma^*$ whether M accepts x.*

Proof of Theorem 48. First, we prove PSPACE-membership. It is not possible to use Algorithm 1 because we should construct the region graph G, which may have a number of states exponential in the number of clocks of the timed automaton $\mathcal{A}$.

However, we can check the reachability of a region r by guessing a path in the region graph from the initial regions to r in polynomial space. This is a fairly standard trick used for showing PSPACE-membership of the reachability problem for classical timed automata with an on-the-fly algorithm [AD94]. The difficult point is that the successor of a given region r can be a neighbour region r' such that $[r] \cap [r'] \neq \varnothing$ provided r' lies in a progress cycle S of G. As we have shown, the entire region r' can be reached from r in $[\![\mathcal{A}]\!]_\Delta$ no matter how small Δ is, by repeating the cycle S. Hence we can add S in one step in the set of reachable states. Such an *acceleration* has been proven correct (Theorem 39) and complete (Theorem 46). So, when guessing the successor of a region r, we must take into account the neighbour regions of r and decide whether they are in a progress cycle or not. This can be checked in PSPACE using the same procedure as for classical timed automata [AD94]. A polynomially bounded part of the memory is reserved for executions of this procedure. Since the content of this part of the memory is not necessary for further computations, it can be reused by subsequent calls and PSPACE-membership follows.

We establish PSPACE-hardness using a reduction of the acceptance problem for LBTM The reduction is similar to [CY91], where a configuration (q, w, i) of a LBTM is encoded by a location (q, i) (that records the control state q and the tape position i) and by the clocks $y_1, \dots, y_{|w|}$, one for each tape cell. We assume without loss of generality that $\Sigma = \{a, b\}$. A clock y_i has the value $y_i = n_a$ if $w_i = a$ and $y_i = n_b > n_a$ if $w_i = b$. This encoding is not preserved by time passing. Thus we need to periodically refresh the values of the clocks. This is done in two phases: (I) resetting the clocks coding a 'b' (by checking $y_i = n_b$), then letting $n_b - n_a$ time unit elapse, and (II) resetting the clock coding an 'a' (by checking $y_i = n_b$ again) and finally letting n_a time unit elapse. During phase (I), the clock that encodes the tape cell pointed by the head, is updated according to the transitions of the LBTM.

We show how to adapt this reduction to the perturbed semantics of timed automata. Due to guards enlargements, equality cannot be tested precisely and the clocks can not store precise values n_a and n_b. However, if Δ is sufficiently small and n_a and n_b are not too close, we can still distinguish clocks coding an 'a' and clocks coding a 'b'. The details of this proof follow.

Let $M = (Q, \Sigma, q_0, q_f, \delta)$ be a LBTM and $x \in \Sigma^*$ be an input word. Let $n = |x|$, $n_a = 3$ and $n_b = 6$. We construct a timed automaton $\mathcal{A}(M, x)$ with $n + 1$ clocks and a location ℓ_f such that M accepts x iff $(\ell_f, v) \in R^0_{\Delta\to 0}$ for some valuation v. Let $\mathcal{A}(M, x) = \langle \mathsf{Loc}, \mathsf{Var}, q_0^{\mathcal{A}}, \mathsf{Lab}, \mathsf{Edg}\rangle$ with:

- $\mathsf{Loc} = \{s_0, s_1, \ell_f\} \cup \{(q, i, j, \varphi, d) \mid q \in Q \wedge 1 \leqslant i \leqslant n \wedge 1 \leqslant j \leqslant n + 1 \wedge \varphi \in \{\mathrm{I}, \mathrm{II}\} \wedge d \in \{\mathsf{left}, \mathsf{right}\}\}$; a location (q, i, j, φ, d) encodes the control state q, the tape position i, the number j of the next clock to be treated, the phase φ of the simulation, and the direction d of the next head movement;
- $\mathsf{Var} = \{y_i \mid 1 \leqslant i \leqslant n\} \cup \{z\}$;
- $q_0^{\mathcal{A}} = (s_0, v_0)$ with $v_0(t) = 0$ for all $t \in \mathsf{Var}$;
- $\mathsf{Lab} = \{\tau\}$;
- The set Edg contains the following edges (we write $\ell \xrightarrow{g,R} \ell'$ when $(\ell, \ell', g, \tau, R) \in \mathsf{Edg}$):

- *Initialization*:
  - $s_0 \xrightarrow{z=3,\{y_i \mid x_i = a\}\cup\{z\}} s_1$
  - $s_1 \xrightarrow{z=3,\{z\}} (q_0, 1, 1, I, \mathsf{left})$

– *Refresh*: for every $(q, i, j, \varphi, d) \in \mathsf{Loc}$ with $j \neq i$ and $j \leqslant n$,
  - $(q, i, j, \varphi, d) \xrightarrow{z \leqslant 0 \wedge y_j \leqslant 4, \varnothing} (q, i, j+1, \varphi, d)$
  - $(q, i, j, \varphi, d) \xrightarrow{z \leqslant 0 \wedge y_j \geqslant 5, \{y_j\}} (q, i, j+1, \varphi, d)$
  - $(q, i, i, \mathrm{II}, d) \xrightarrow{z \leqslant 0 \wedge y_i \leqslant 4, \varnothing} (q, i, i+1, \mathrm{II}, d)$
  - $(q, i, i, \mathrm{II}, d) \xrightarrow{z \leqslant 0 \wedge y_i \geqslant 5, \{y_i\}} (q, i, i+1, \mathrm{II}, d)$

– *Execution*: for every $q \in Q$, $1 \leqslant i \leqslant n$, $d \in \{\mathsf{left}, \mathsf{right}\}$, and for every transition $(q, \sigma, \sigma', d', q') \in E$,
  - If $(\sigma, \sigma') = (a, a)$ then $(q, i, i, \mathrm{I}, d) \xrightarrow{z \leqslant 0 \wedge y_i \leqslant 4, \varnothing} (q', i, i+1, \mathrm{I}, d')$
  - If $(\sigma, \sigma') = (a, b)$ then $(q, i, i, \mathrm{I}, d) \xrightarrow{z \leqslant 0 \wedge y_i \leqslant 4, \{y_i\}} (q', i, i+1, \mathrm{I}, d')$
  - If $(\sigma, \sigma') = (b, a)$ then $(q, i, i, \mathrm{I}, d) \xrightarrow{z \leqslant 0 \wedge y_i \geqslant 5, \varnothing} (q', i, i+1, \mathrm{I}, d')$
  - If $(\sigma, \sigma') = (b, b)$ then $(q, i, i, \mathrm{I}, d) \xrightarrow{z \leqslant 0 \wedge y_i \geqslant 5, \{y_i\}} (q', i, i+1, \mathrm{I}, d')$

– *Phase change*: for every $q \in Q$, $1 \leqslant i \leqslant n$, $j = n+1$ and $d \in \{\mathsf{left}, \mathsf{right}\}$,
  - $(q, i, n+1, \mathrm{I}, d) \xrightarrow{z=3, \{z\}} (q, i, 1, \mathrm{II}, d)$
  - $(q, i, n+1, \mathrm{II}, \mathsf{left}) \xrightarrow{z=3, \{z\}} (q, i-1, 1, \mathrm{I}, \mathsf{left})$
  - $(q, i, n+1, \mathrm{II}, \mathsf{right}) \xrightarrow{z=3, \{z\}} (q, i+1, 1, \mathrm{I}, \mathsf{right})$

– *Termination*: for every $1 \leqslant i \leqslant n$, $d \in \{\mathsf{left}, \mathsf{right}\}$,
  - $(q_f, i, 1, \mathrm{I}, d) \xrightarrow{\mathsf{true}, \varnothing} \ell_f$

After the *initialization step*, the automaton is in the location $(q_0, 1, 1, I, \mathsf{left})$ and we have the following relations between the tape content w and the clocks $y_1, \ldots, y_n$ when $z = 0$:

$$\begin{cases} 3 - \Delta \leqslant y_i \leqslant 3 + \Delta & \text{if } w_i = a \\ 6 - 2\Delta \leqslant y_i \leqslant 6 + 2\Delta & \text{if } w_i = b \end{cases}$$

After executing one transition $(q, \sigma, \sigma', d', q')$ of M, let w' be the new tape content (w' differs from w by at most one symbol). If we simulate that transition by the *refresh* steps, the *execution* step, and the *phase changes*, it is easy to check that in location $(q, i, 1, I, d)$, when $z = 0$ we have:

$$\begin{cases} 3 - 2\Delta \leqslant y_i \leqslant 3 + \Delta & \text{if } w'_i = a \\ 6 - 3\Delta \leqslant y_i \leqslant 6 + 2\Delta & \text{if } w'_i = b \end{cases} \tag{11}$$

Note that two clocks coding the same symbol are not necessarily equal (however, their difference is bounded by Δ). The reader can check that after having executed a second transition of M, there is no accumulation of the imprecisions and the conditions (11) still hold. Hence, provided Δ is sufficiently small (in fact $\Delta < 1/2$), the automaton $\mathcal{A}(M, x)$ will correctly distinguish clocks coding 'a' from clocks coding 'b' for any number of transitions, and thus simulate faithfully the execution of M on x. It is now easy to see that the location ℓ_f is reachable in $R^0_{\Delta \to 0}$ iff ℓ_f is reachable in $[\![\mathcal{A}]\!]^0_0$ iff M accepts x. This concludes the proof since our construction is polynomial in the size of M and x. ∎

References

[AAB00] Aurore Annichini, Eugène Asarin, and Ahmed Bouajjani. Symbolic techniques for parametric reasoning about counter and clock systems. In *Proc. 12th Int. Conf. Computer Aided Verification (CAV 2000)*, pages 419–434, 2000.

[AB01] Eugène Asarin and Ahmed Bouajjani. Perturbed Turing machines and hybrid systems. In *Proc. 16th Annual Symposium on Logic in Computer Science (LICS)*, pages 269–278. IEEE Computer Society Press, 2001.

[ACD+92] Rajeev Alur, Costas Courcoubetis, David L. Dill, Nicolas Halbwachs, and Howard Wong-Toi. An implementation of three algorithms for timing verification based on automata emptiness. In *Proc. 13th IEEE Real-time Systems Symposium*, pages 157–166. IEEE Comp. Soc. Press, 1992.

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

[AFM+02] Tobias Amnell, Elena Fersman, Leonid Mokrushin, Paul Pettersson, and Wang Yi. Times: A tool for modelling and implementation of embedded systems. In Joost-Pieter Katoen and Perdita Stevens, editors, *Proc. 8th Int. Conference Tools and Algorithms for the Construction and Analysis of Systems (TACAS'02)*, volume 2280 of *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*, 9, 2003.

[AIK+03] Rajeev Alur, Franjo Ivancic, Jesung Kim, Insup Lee, and Oleg Sokolsky. Generating embedded software from hierarchical hybrid models. In *Proc. 2003 Conf. Languages, Compilers, and Tools for Embedded Systems (LCTES'03)*, pages 171–182, 2003.

[ALM05] Rajeev Alur, Salvatore La Torre, and P. Madhusudan. Perturbed timed automata. In *Proc. 8th Int. Workshop Hybrid Systems: Computation and Control (HSCC'05)*, volume 3414 of *Lecture Notes in Computer Science*, pages 70–85. Springer, 2005.

[AMPS98] Eugène Asarin, Oded Maler, Amir Pnueli, and Joseph Sifakis. Controller synthesis for timed automata. In *Proc. System Structure and Control*. Elsevier, 1998.

[AT04] Manindra Agrawal and P. S. Thiagarajan. Lazy rectangular hybrid automata. In *Proc. of HSCC 04: Hybrid Systems—Computation and Control*, Lecture Notes in Computer Science 2993, pages 1–15. Springer-Verlag, 2004.

[AT05] Karine Altisen and Stavros Tripakis. Implementation of timed automata: an issue of semantics or modeling? In *Proc. 3rd Int. Conf. Formal Modelling and Analysis of Timed Systems (FORMATS'05)*, Lecture Notes in Computer Science. Springer, 2005.

[BC05] Patricia Bouyer and Fabrice Chevalier. On conciseness of extensions of timed automata. *Journal of Automata, Languages and Combinatorics*, 2005. To appear.

[BDL+06] Gerd Behrmann, Alexandre David, Kim Guldstrand Larsen, John Håkansson, Paul Pettersson, Wang Yi, and Martijn Hendriks. Uppaal 4.0. In *QEST*, pages 125–126, 2006.

[BM83] Bernard Berthomieu and Miguel Menasche. An enumerative approach for analyzing time Petri nets. In R.E.A. Mason, editor, *Information Processing 83 – Proceedings of the 9th IFIP World Computer Congress*, pages 41–46. North-Holland/IFIP, September 1983.

[BMR06] Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust model-checking of linear-time properties in timed automata. In José R. Correa, Alejandro Hevia, and Marcos Kiwi, editors, *Proceedings of the 7th Latin American Symposium on Theoretical INformatics (LATIN'06)*, volume 3887 of *Lecture Notes in Computer Science*, pages 238–249. Springer-Verlag, March 2006.

[BMR08] Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust analysis of timed automata *via* channel machines. In Roberto Amadio, editor, *Proceedings of the 11th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'08)*, Lecture Notes in Computer Science, Budapest, Hungary, March-April 2008. Springer. To appear.

[CGP99] Edmund Clarke, Orna Grumberg, and Doron Peled. *Model Checking*. MIT Press, 1999.

[CHR91] Zhou Chaochen, C. A. R. Hoare, and Anders P. Ravn. A calculus of durations. *Information Processing Letters*, 40(5):269–276, 1991.

[CHR02] Franck Cassez, Thomas A. Henzinger, and Jean-François Raskin. A comparison of control problems for timed and hybrid systems. In *Proc. 5th Int. Workshop Hybrid Systems: Computation and Control (HSCC'02)*, volume 2289 of *Lecture Notes in Computer Science*, pages 134–148. Springer-Verlag, 2002.

[CHS93] Zhou Chaochen, Michael R. Hansen, and Peter Sestoft. Decidability and undecidability results for duration calculus. In *In Proc. of STACS 93: Symposium on Theoretical Aspects of Computer Science*, Lecture Notes in Computer Science 665, pages 58–68. Springer-Verlag, 1993.

[CY91] Costas Courcoubetis and Mihalis Yannakakis. Minimum and maximum delay problems in real-time systems. In *Proc. 3rd Int. Workshop Computer Aided Verification (CAV'91)*, volume 575 of *Lecture Notes in Computer Science*, pages 399–409. Springer-Verlag, 1991.

[DDMR04] Martin De Wulf, Laurent Doyen, Nicolas Markey, and Jean-François Raskin. Robustness and implementability of timed automata. In Y. Lakhnech and S. Yovine, editors, *Proceedings of the Joint Conferences Formal Modelling and Analysis of Timed Systems (FORMATS'04) and Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'04)*, volume 3253 of *Lecture Notes in Computer Science*, pages 118–133, Grenoble, France, September 2004. Springer.

[DDR05] Martin De Wulf, Laurent Doyen, and Jean-François Raskin. Almost ASAP semantics: From timed models to timed implementations. *Formal Aspects of Computing*, 17(3):319–341, October 2005.

[Die99] Henning Dierks. *Specification and Verification of Polling Real-Time Systems*. PhD thesis, University of Oldenburg, July 1999.

[Die01] Henning Dierks. PLC-automata: a new class of implementable real-time automata. *Theoretical Computer Science*, 253(1):61–93, 2001.

[Dil90] David Dill. Timing assumptions and verification of finite-state concurrent systems. In *Proc. 1st Int. Workshop Automatic Verification Methods for Finite State Systems (CAV'89)*, volume 407 of *Lecture Notes in Computer Science*, pages 197–212. Springer-Verlag, 1990.

[Dim07] Cătălin Dima. Dynamical properties of timed automata revisited. In *Proc. of FORMATS 07: Formal Modeling and Analysis of Timed Systems*, Lecture Notes in Computer Science 4763, pages 130–146. Springer-Verlag, 2007.

[DK06] Conrado Daws and Piotr Kordy. Symbolic robustness analysis of timed automata. In *Proc. of FORMATS 06: Formal Modeling and Analysis of Timed Systems*, Lecture Notes in Computer Science 4202, pages 143–155. Springer-Verlag, 2006.

[Frä99] Martin Fränzle. Analysis of hybrid systems: An ounce of realism can save an infinity of states. In *CSL*, Lecture Notes in Computer Science 1683, pages 126–140. Springer-Verlag, 1999.

[GHJ97] Vineet Gupta, Thomas A. Henzinger, and Radha Jagadeesan. Robust timed automata. In O. Maler, editor, *Proc. Int. Workshop Hybrid and Real-Time Systems (HART'97)*, volume 1201 of *Lecture Notes in Computer Science*, pages 331–345. Springer Verlag, March 1997.

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

[HNSY92] Thomas A. Henzinger, Xavier Nicollin, Joseph Sifakis, and Sergio Yovine. Symbolic model checking for real-time systems. In *Proc. 7th Annual Symposium Logic in Computer Science (LICS'92)*, pages 394–406. IEEE Comp. Soc. Press, 1992.

[HRSV01] Thomas Hune, Judi Romijn, Marielle Stoelinga, and Frits W. Vaandrager. Linear parametric model checking of timed automata. In *Proc. 7th Int. Conf. Tools and Algorithms for Construction and Analysis of Systems (TACAS'01)*, pages 189–203, 2001.

[Mil80] Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, 1980.

[MPS95] Oded Maler, Amir Pnueli, and Joseph Sifakis. On the synthesis of discrete controllers for timed systems (an extended abstract). In *STACS*, pages 229–242, 1995.

[Pur98] Anuj Puri. Dynamical properties of timed automata. In *Proc. 5th Int. Symposium Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'98)*, volume 1486 of *Lecture Notes in Computer Science*, pages 210–227. Springer, 1998.

[Pur00] Anuj Puri. Dynamical properties of timed automata. *Discrete Event Dynamic Systems*, 10(1-2):87–113, Jan 2000.

[SF07] Mani Swaminathan and Martin Fränzle. A symbolic decision procedure for robust safety of timed systems. In *Proceedings of the 14th International Symposium on Temporal Representation and Reasoning (TIME'07)*, page 192. IEEE Comp. Soc. Press, June 2007.

[Yov96] Sergio Yovine. Model checking timed automata. In *European Educational Forum: School on Embedded Systems*, pages 114–152, 1996.