Time-bounded Reachability for Hybrid Automata: Complexity and Fixpoints

Thomas Brihaye*  Laurent Doyen†  Gilles Geeraerts%  Joël Ouaknine†  Jean-François Raskin%  James Worrell†  

*: Université de Mons, Belgium  
+: LSV, ENS Cachan & CNRS, France  
%: Université Libre de Bruxelles, Belgium  
†: Oxford University Computing Lab., UK  

Abstract

In this paper, we study the *time-bounded reachability problem* for rectangular hybrid automata with non-negative rates (RHA $\geq 0$). This problem was recently shown to be decidable [5] (even though the *unbounded* reachability problem for even very simple classes of hybrid automata is well-known to be undecidable). However, [5] does not provide a precise characterisation of the complexity of the time-bounded reachability problem. The contribution of the present paper is threefold. First, we provide a new NE$\text{XP}T$IME algorithm to solve the timed-bounded reachability problem on RHA $\geq 0$. This algorithm improves on the one of [5] by at least one exponential. Second, we show that this new algorithm is optimal, by establishing a matching lower bound: time-bounded reachability for RHA $\geq 0$ is therefore NE$\text{XP}T$IME-complete. Third, we extend these results in a practical direction, by showing that we can effectively compute fixpoints that characterise the sets of states that are reachable (resp. co-reachable) within $T$ time units from a given starting state.

1 Introduction

*Hybrid systems* form a general class of systems that mix *continuous* and *discrete* behaviors. Examples of hybrid systems abound in our everyday life, particularly in applications where an (inherently discrete) computer system must interact with a continuous environment. The need for modeling hybrid systems is obvious, together with methods to analyse those systems.

*Hybrid automata* are arguably among the most prominent families of models for hybrid systems [7]. Syntactically, a hybrid automaton is a finite automaton (to model the discrete part of the system) augmented with a finite set of real-valued variables (to model the continuous part of the system). Those variables evolve with time elapsing, at a rate which is given by a flow function that depends on the current location of the automaton. The theory of hybrid automata has been well developed for about 20 years,
and tools to analyse them are readily available, see for instance HYTECH \cite{8,9} and PHAVER \cite{6}.

Hybrid automata are thus a class of powerful models, yet their high expressiveness comes at a price, in the sense that the undecidability barrier is rapidly hit. Simple reachability properties are undecidable even for the restricted subclass of stopwatch automata, where the rate of growth of each variable stays constant in all locations and is restricted to either 0 or 1 (see \cite{10} for a survey).

On the other hand, a recent and successful line of research in the setting of timed automata has outlined the benefits of investigating timed-bounded variants of classical properties \cite{12,14}. For instance, while language inclusion is, in general undecidable for timed automata, it becomes decidable when considering only executions of bounded duration \cite{14}.

In a recent work \cite{5} we have investigated the decidability of time-bounded reachability for rectangular hybrid automata (i.e., is a given state reachable by an execution of duration at most $T$ for a given $T$). We have shown that time-bounded reachability is decidable for rectangular hybrid automata with non-negative rates (RHA $\geq 0$), while it is well-known \cite{10} that (plain, time unbounded) reachability is not for this class. We have also shown that the decidability frontier is quite sharp in the sense that time-bounded reachability becomes undecidable once we allow either diagonal constraints in the guards or negative rates.

To obtain decidability of time-bounded reachability for RHA $\geq 0$, we rely, in \cite{5}, on a contraction operator that applies to runs, and allows to derive, from any run of duration at most $T$ of an RHA $\geq 0$ $H$, an equivalent run that reaches the same state, but whose length (in terms of number of discrete transitions) is uniformly bounded by a function $F$ of the size of $H$ and $T$. Hence, deciding reachability within $T$ time units reduces to exploring runs of bounded lengths only, which is feasible algorithmically (see \cite{5} for the details). However, this previous work does not contain a precise characterisation of the complexity of time-bounded reachability. Clearly, an upper bound on the complexity depends on the bound $F$ on the length of the runs that need to be explored.

In the present work, we revisit and extend our previous results \cite{5} in several directions, both from the theoretical and the practical point of view. First, we completely revisit the definition of the contraction operator and obtain a new operator that allows to derive a singly exponential upper bound on the lengths of the runs that need to be considered, while the operator in \cite{5} yields an upper bound that is at least doubly exponential. Our new contraction operator thus provides us with an NExpTime algorithm that improves on the algorithm of \cite{5} by at least one exponential. Second, we show that this new algorithm is optimal, by establishing a matching lower bound. Hence, time-bounded reachability for RHA $\geq 0$ is NExpTime-complete. Third, we extend those results towards more practical concerns, by showing that we can effectively compute fixpoints that characterise the set of states that are reachable (resp. co-reachable) within $T$ time units, from a given state. The time needed to compute them is at most doubly exponential in the size of the RHA $\geq 0$ and the bound $T$. Fourth, we apply those ideas to two examples of RHA $\geq 0$ for which the classical (time-unbounded) forward and backward fixpoints do not terminate. We show that, in those examples, the sets of states that are time-bounded reachable is computable in practice, for values of the time bound that allow us to derive meaningful properties.

This brief summary of the results outlines the structure of the paper. Remark that, by lack of space, some more technical proofs have been moved to the appendix.
2 Definitions

Let $\mathcal{I}$ be the set of intervals of real numbers with endpoints in $\mathbb{Z} \cup \{\infty, +\infty\}$. Let $X$ be a set of continuous variables, and let $\tilde{X} = \{\dot{x} \mid x \in X\}$ be the set of dotted variables, corresponding to variable first derivatives. A rectangular constraint over $X$ is an expression of the form $x \in I$ where $x$ belongs to $X$ and $I$ to $\mathcal{I}$. A diagonal constraint over $X$ is a constraint of the form $x - y \sim c$ where $x, y$ belong to $X$, $c$ to $\mathbb{Z}$, and $\sim$ is in $\{<, \leq, =, \geq, >\}$. Finite conjunctions of diagonal and rectangular constraints over $X$ are called guards, over $\tilde{X}$ they are called rate constraints. A guard or rate constraint is rectangular if all its constraints are rectangular. We denote by $\mathcal{G}(X)$ and $\mathcal{R}(X)$ respectively the sets of guards and rate constraints over $X$.

Linear, rectangular and singular hybrid automata A linear hybrid automaton (LHA) is a tuple $\mathcal{H} = (X, Loc, Edges, Rates, Inv, Init)$ where $X = \{x_1, \ldots, x_{|X|}\}$ is a finite set of continuous variables; $Loc$ is a finite set of locations; $Edges \subseteq Loc \times \mathcal{G}(X) \times 2^X \times Loc$ is a finite set of edges; $Rates : Loc \mapsto \mathcal{R}(X)$ assigns to each location a constraint on the possible variable rates; $Inv : Loc \mapsto \mathcal{G}(X)$ assigns an invariant to each location; and $Init \subseteq Loc$ is a set of initial locations. For an edge $e = (\ell, g, Y, \ell')$, we denote by $\text{src}(e)$ and $\text{trg}(e)$ the location $\ell$ and $\ell'$ respectively. $g$ is called the guard of $e$ and $Y$ is the reset set of $e$. In the sequel, we denote by $r_{\text{max}}$ and $r_{\text{max}}$ the maximal constant occurring respectively in the constraints of $\text{Rates}(\ell) \mid \ell \in Loc$ and of $\text{Rates}(\ell) \mid \ell \in Loc \cup \{g \mid \exists (\ell, g, Y, \ell') \in Edges\}$.

An LHA is non-negative rate if for all variables $x$, for all locations $\ell$, the constraint $Rates(\ell)$ implies that $\dot{x}$ must be non-negative. A rectangular hybrid automaton (RHA) is a linear hybrid automaton in which all guards, rates, and invariants are rectangular. In the case of RHA, we view rate constraints as functions $Rates : \text{Loc} \times X \to \mathcal{I}$ that associate with each location $\ell$ and each variable $x$ an interval of possible rates $Rates(\ell)(x)$. A singular hybrid automaton (SHA) is an RHA s.t. for all locations $\ell$ and for all variables $x$: $Rates(\ell)(x)$ is a singleton. We use the shorthands $RHA^{\geq 0}$ and $SHA^{\geq 0}$ for non-negative rates RHA and SHA respectively.

LHA semantics A valuation of a set of variables $X$ is a function $\nu : X \mapsto \mathbb{R}$. We denote by $0$ the valuation that assigns $0$ to each variable.

Given an LHA $\mathcal{H} = (X, Loc, Edges, Rates, Inv, Init, X)$, a state of $\mathcal{H}$ is a pair $(\ell, \nu)$, where $\ell \in Loc$ and $\nu$ is a valuation of $X$. The semantics of $\mathcal{H}$ is defined as follows. Given a state $s = (\ell, \nu)$ of $\mathcal{H}$, an edge step $(\ell, \nu) \xrightarrow{e} (\ell', \nu')$ can occur and change the state to $(\ell', \nu')$ if $e = (\ell, g, Y, \ell') \in Edges$, $\nu \models g$, $\nu'(x) = \nu(x)$ for all $x \notin Y$, and $\nu'(x) = 0$ for all $x \in Y$: given a time delay $t \in \mathbb{R}_+$, a continuous time step $(\ell, \nu) \xrightarrow{t} (\ell, \nu')$ can occur and change the state to $(\ell, \nu')$ if there exists a vector $r = (r_1, \ldots, r_{|X|})$ such that $r \models \text{Rates}(\ell)$, $\nu' = \nu + (r \cdot t)$, and $\nu + (r \cdot t') \models \text{Inv}(\ell)$ for all $0 \leq t' \leq t$.

A path in $\mathcal{H}$ is a finite sequence $e_1, e_2, \ldots, e_n$ of edges such that $\text{trg}(e_i) = \text{src}(e_{i+1})$ for all $1 \leq i \leq n - 1$. A timed path of $\mathcal{H}$ is a finite sequence of the form $\pi = (t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$, such that $e_1, \ldots, e_n$ is a path in $\mathcal{H}$ and $t_i \in \mathbb{R}_+$ for all $0 \leq i \leq n$. For all $k, \ell$, we denote by $\pi[k : \ell]$ the maximal portion $(t_1, e_1), (t_{i+1}, e_{i+1}), \ldots, (t_j, e_j)$ of $\pi$ s.t. $\{i, i + 1, \ldots, j\} \subseteq [k, \ell]$ (remark that the interval $[k, \ell]$ could be empty, then $\pi[k : \ell]$ is empty too). Given a timed path $\pi = (t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$ of an SHA, we let $\text{Effect}(\pi) = \sum_{i=1}^{n} \text{Rates}(\ell_{i-1}) \cdot t_i$ be the effect of $\pi$ (where $\ell_i = \text{src}(e_i)$ for $1 \leq i \leq n$).
A run in $\mathcal{H}$ is a sequence $s_0, (t_1, e_1), s_1, (t_2, e_2), \ldots, (t_n, e_n), s_n$ such that:

- $(t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$ is a timed path in $\mathcal{H}$,
- for all $0 \leq i < n$, there exists a state $s_i'$ of $\mathcal{H}$ with $s_i \xrightarrow{t_{i+1}, e_{i+1}} s_i' \xrightarrow{t_{i+1}, e_{i+1}} s_{i+1}$.

Given a run $\rho = s_0, (t_1, e_1), \ldots, s_n$, let first $(\rho) = s_0 = (\ell_0, \nu_0)$, last $(\rho) = s_n$, duration $(\rho) = \sum_{i=1}^{n} t_i$, and $|\rho| = n + 1$. We say that $\rho$ is $T$-time-bounded (for $T \in \mathbb{N}$) if duration $(\rho) \leq T$. Given two runs $\rho = s_0, (t_1, e_1), \ldots, (t_n, e_n), s_n$ and $\rho' = s_0', (t_1', e_1'), \ldots, (t_k', e_k'), s_k'$ with $s_n = s_n'$, we let $\rho \cdot \rho'$ denote the run $s_0, (t_1, e_1), \ldots, (t_n, e_n), s_n, (t_1', e_1'), \ldots, (t_k', e_k'), s_k'$.

Note that a unique timed path $\text{TPath}(\rho) = (t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$, is associated with each run $\rho = s_0, (t_1, e_1), s_1, \ldots, (t_n, e_n), s_n$. Hence, we sometimes abuse notation and denote a run $\rho$ with first $(\rho) = s_0$, last $(\rho) = s$ and $\text{TPath}(\rho) = \pi$ by $s_0 \xrightarrow{\pi} s$. The converse however is not true: given a timed path $\pi$ and an initial state $s_0$, it could be impossible to build a run starting from $s_0$ and following $\pi$ because some guards or invariants along $\pi$ might be violated. However, if such a run exists it is necessarily unique when the automaton is singular. In that case, we denote by $\text{Run}(s_0, \pi)$ the function that returns the unique run $\rho$ such that first $(\rho) = s_0$ and $\text{TPath}(\rho) = \pi$ if it exists, and $\bot$ otherwise. Remark that, when consider an SHA: if $\rho = (\ell_0, \nu_0) \xrightarrow{\pi} (\ell_n, \nu_n)$ is a run, then for all $x$ that is not reset along $\rho$: $\nu_n(x) = \nu_0(x) + \text{Effect}(\pi)(x)$.

**Time-bounded reachability problem for LHA** While the reachability problem asks whether there exists a run reaching a given goal location, we are only interested in runs having bounded duration.

**Problem 1 (Time-bounded reachability problem)** Given an LHA $\mathcal{H} = (X, \text{Loc, Edges, Rates, Inv, Init})$, a location $\text{Goal} \in \text{Loc}$ and a time bound $T \in \mathbb{N}$, the time-bounded reachability problem is to decide whether there exists a finite run $\rho = (\ell_0, 0) \xrightarrow{\pi} (\text{Goal, } \cdot)$ of $\mathcal{H}$ with $\ell_0 \in \text{Init}$ and duration $(\rho) \leq T$.

This problem is known to be decidable [5] for RHA$^{\geq 0}$, but its exact complexity is, so far, unknown. We prove in Section 3 (thanks to the results of Section 4) that it is NEXPTIME-complete. This problem is known to become undecidable once we allow either diagonal constraints in the guards, or negative and positive rates to occur in the LHA [5].

A more general problem that is relevant in practice, is to compute a symbolic representation of all the states that are reachable in at most $T$ time units. Here, by ‘symbolic representation’ we mean a finite representation of the set of states that can be manipulated algorithmically. This problem, together with the definition of such a a symbolic representation, will be addressed in Section 5.

Let us illustrate, by means of the RHA$^{\geq 0}$ $\mathcal{H}$ in Fig. 1 the difficulties encountered when computing the reachable states of a RHA$^{\geq 0}$. Let us characterise the set $\text{Reach}_H(s_0)$ of all states of the form $(\ell_1, \nu)$ that are reachable from $s_0$. It is easy to see that $\text{Reach}_H(s_0) = \{(\ell_1, (0, 1/n)) \mid n \in \mathbb{N}_0\}$. Moreover, observe that, for all $n \in \mathbb{N}_0$, $(\ell_1, (0, 1/n))$ is reachable from $s_0$ by one and only one run, of duration $(n - 1) + \frac{1}{n}$, and that the number of bits necessary to encode those states grows linearly with the length of the run. This examples shows that finding an adequate, compact and effective representation (such as regions in the case of Timed Automata) [2] for the set of reachable of an RHA$^{\geq 0}$ is not trivial (and, in full generality, impossible because reachability...
is undecidable for this class). Nevertheless, in Section 5, we show that, in an RHA\textsuperscript{≥0}, an effective representation of the set of states that are reachable within \(T\) time units can be computed.

## 3 Contracting runs

In this section, we describe a contraction operator. Given an RHA \(H\), and one of its timed paths \(\pi\) of arbitrary length but of duration \(\leq T\), the contraction operator builds a timed path \(\text{Cnt}^* (\pi)\) that reaches the same state as \(\pi\), but whose size is uniformly bounded by a function of \(T\), and of the size of \(H\). This operator is central to prove correctness of the algorithms for time-bounded reachability in sections 4 and 5. Since Problem \(\Pi\) is undecidable if both positive and negative rates are allowed \([5]\), we restrict our attention to RHA with non negative rates. Moreover, for the sake of clarity, all the results presented in this section are limited to singular hybrid automata, but they extend easily to RHA\textsuperscript{≥0} as we will see later. Thus, from now one, we fix an SHA\textsuperscript{≥0}\(H = (X, \text{Loc}, \text{Edges}, \text{Rates}, \text{Inv}, \text{Init})\).

### Self loops

The first step of our construction consists in adding, on each location \(\ell\) of \(H\), a self-loop \((\ell, \text{true}, \emptyset, \ell)\). The resulting SHA\textsuperscript{≥0} is called \(H'\). Those self-loops allow to split runs of \(H'\) into portions of arbitrary small delays, because if \(H'\) admits a run of the form \((\ell, \nu), (t_1 + t_2, e), s\), it also admits the run \((\ell, \nu), (t_1, e'), (\ell, \nu'), (t_2, e), s\), where \(e'\) is the self loop on \(\ell\). Yet, this construction preserves (time-bounded) reachability:

**Lemma 1** Every run of \(H\) is also a run of \(H'\). Conversely, if \(H'\) admits a run \(\rho'\) with first \((\rho') = s_1\) and last \((\rho') = s_2\), then \(H\) admits a run \(\rho\) with \(\rho\) first \((\rho) = s_1\), last \((\rho) = s_2\), duration \((\rho) = \text{duration}(\rho')\) and \(|\rho| \leq |\rho'|\). Moreover, for each run \(\rho\) of \(H\), there exists a run \(\rho' = \rho_1 \cdot \rho_2 \cdots \rho_n\) of \(H'\) s.t. \(n \leq \text{duration}(\rho) \times r_{\text{max}} + 1\), first \((\rho) = \text{first}(\rho')\), last \((\rho) = \text{last}(\rho')\), duration \((\rho) = \text{duration}(\rho')\) and, for all \(1 \leq i \leq n\): duration \((\rho_i) < \frac{1}{r_{\text{max}}}\).

### Hybrid automaton with regions

Let us describe a second construction that applies to the syntax of the hybrid automaton, and consists, roughly speaking, in encoding the integral part of the variable valuations in the locations. Let \(\text{Reg}(c_{\text{max}}) = \{[a, a], (a - 1, a) \mid a \in \{1, \ldots, c_{\text{max}}\}\} \cup \{0^-, 0^+, (c_{\text{max}}, +\infty)\}\) be the set of regions, and further let \(\text{Reg}(c_{\text{max}}, X)\) denote the set of all functions \(r : X \rightarrow \text{Reg}(c_{\text{max}})\) that assign a region to each variable. By abuse of language, we sometimes call regions elements of \(\text{Reg}(c_{\text{max}}, X)\) too. Remark that the definition of \(\text{Reg}(c_{\text{max}}, X)\) differs from the classical regions \([2]\) by the absence of \([0, 0]\) which is replaced by two

\[
\begin{align*}
\ell_0 & \xrightarrow{\ell, 2} x = 1 \\
\ell_1 & \xrightarrow{\ell, 0} \begin{array}{l}
x := 0 \\
y = 1 \\
y := 0
\end{array}
\end{align*}
\]

Figure 1: A simple hybrid automaton.
symbols: $0^-$ and $0^+$, and by the fact that no information is retained about the relative values of the fractional parts of the variables. The difference between $0^-$ and $0^+$ is elucidated later (see Lemma 3). When testing for membership to a region, $0^+$ and $0^-$ should be interpreted as $[0, 0]$, i.e., $v \in 0^+$ and $v \in 0^-$ hold iff $v = 0$. Given a valuation $\nu$ of the set of variable $X$, and $r \in \text{Reg}(c_{\text{max}}, X)$, we let $\nu \in r$ iff $\nu(x) \in r(x)$ for all $x$, and, provided that $\nu > 0$, we denote by $[\nu]$ the (unique) element from $\text{Reg}(c_{\text{max}}, X)$ s.t. $\nu \in [\nu]$. Remark that for all sets of variable $X$ and all maximal constants $c_{\text{max}}$, $|\text{Reg}(c_{\text{max}}, X)| \leq (2 \times (c_{\text{max}} + 1))^{|X|}$. Let $r_1$ and $r_2$ be two regions in $\text{Reg}(c_{\text{max}}, X)$, and let $v : X \mapsto \mathbb{R}$ be a function assigning a rate $v(x)$ to each variable $x$. Then, we say that $r_2$ is a time successor of $r_1$ under $v$ (written $r_1 \lessdot_{\text{ts}} r_2$) iff there are $\nu_1 \in r_1$, $\nu_2 \in r_2$ and a time delay $t$ s.t. $\nu_2 = \nu_1 + t \cdot v$. Remark that, by this definition, we can have $r_1 \lessdot_{\text{ts}} r_2$, $r_1(x) = 0^-$ and $r_2(x) = 0^+$ for some clock $x$ (for instance, if $v(x) = 0$).

Let us now explain how we label the locations of $\mathcal{H}'$ by regions. We let $R(\mathcal{H}') = (X, \text{Loc}', \text{Edges}', \text{Rates}', \text{Inv}', \text{Init}')$ be the SHA$^{\geq 0}$ where:

- $\text{Loc}' = \text{Loc} \times \text{Reg}(c_{\text{max}}, X)$
- for all $(\ell, r) \in \text{Loc}'$: $\text{Rates}'(\ell, r) = \text{Rates}(\ell)$
- for all $(\ell, r) \in \text{Loc}'$: $\text{Inv}(\ell, r) = \text{Inv}(\ell) \land \bigwedge_{x \in X \land r(x) = 0} x = 0$
- There is an edge $e' = ((\ell, r), g \land x \in r'' \land g_0, (\ell', r'))$ in $\text{Edges}'$ iff there is an edge $e = (\ell, g, Y, \ell')$ in $\text{Edges}$ and a region $r''$ s.t. $r \lessdot_{\text{Inv}(\ell)} r''$, for all $x \notin Y$: $r''(x) = r'(x)$, for all $x \in Y$: $r''(x) \in \{0^-, 0^+\}$ and $g_0 = \bigwedge_{x \in X} g_0(x)$ where:

$$\forall x \in X : g_0(x) = \begin{cases} x = 0 \quad \text{if } r(x) = 0^- \\ x > 0 \quad \text{if } r(x) = 0^+ \\ \text{true otherwise } \end{cases}$$ (1)

in this case, we say that $e$ is the (unique) edge of $\mathcal{H}'$ corresponding to $e'$. Symmetrically, $e'$ is the only edge corresponding to $e$ between locations $(\ell, r)$ and $(\ell', r')$.

- $\text{Init}' = \text{Init} \times \{0^-, 0^+\}^X$

It is easy to see that this construction incurs an exponential blow up in the number of locations. More precisely:

$$|\text{Loc}'| \leq |\text{Loc}| \times |\text{Reg}(c_{\text{max}}, X)|$$

$$= |\text{Loc}| \times (2 \times (c_{\text{max}} + 1))^{|X|}$$ (2)

Let us prove that this construction preserves reachability of states:

**Lemma 2** Let $s = (\ell, \nu)$ and $s' = (\ell', \nu')$ be two states of $\mathcal{H}'$. Then, $\mathcal{H}'$ admits a run $\rho$ with first $(\rho) = s$ and last $(\rho) = s'$ iff there are $r$ and $r'$ s.t. $R(\mathcal{H}')$ admits a run $\rho'$ with first $(\rho') = ((\ell, r), \nu)$, last $(\rho') = ((\ell', r'), \nu')$, duration $(\rho) = \text{duration}(\rho')$ and $|\rho| = |\rho'|$.

Intuitively, the regions that label locations in $R(\mathcal{H}')$ are intended to track the region to which each variable belongs when entering the location. However, in the case where a variable $x$ enters a location with value 0, we also need to remember whether $x$ is
still null when crossing the next edge (for reasons that will be made clear later). This explains why we have two regions, $0^-$ and $0^+$, corresponding to value 0. They encode respectively the fact that the variable is null (strictly positive) when leaving the location.

Formally, we say that a run $\rho = ((t_0,r_0), (t_1,e_1), (t_2,r_2), \ldots, (t_n,e_n))$, $((\ell_0, r_0), (\ell_1, e_1), (\ell_2, r_2), \ldots, (\ell_n, e_n))$, $((\ell_0, r_0), (\ell_1, e_1))$ of $\mathcal{R}(\mathcal{H'})$ is region consistent iff (i) for all $0 \leq i \leq n$: $\nu_i \in r_i$ and (ii) for all $0 \leq i \leq n - 1$: $\nu_i \in r_i$ and $\nu_{i+1} \in r_{i+1}$.

Then, it is easy to see that the construction of $\mathcal{R}(\mathcal{H'})$ guarantees that all runs are region consistent:

**Lemma 3** All runs of $\mathcal{R}(\mathcal{H'})$ are region consistent.

The contraction operator we are about to describe preserves reachability of states when applied to carefully selected run portions only. Those portions are obtained by splitting several times a complete run into sub-runs, that we categorise in 4 different types.

**Type-0 and type-1 runs** The notion of type-0 run relies on the fact that each $T$-time bounded run of $\mathcal{H'}$ (hence of $\mathcal{R}(\mathcal{H'})$) corresponds to a run $\rho'$ that can be split into at most $T \times \max + 1$ portions of duration $< \frac{1}{\max}$ (see Lemma 1). A run $\rho$ of $\mathcal{R}(\mathcal{H'})$ is called a type-0 run iff there are $\rho_0, \rho_1, \ldots, \rho_k$ s.t. $\rho = \rho_0 \cdot \rho_1 \cdots \rho_k$, and for all $0 \leq i \leq k$: duration $(\rho_i) < \frac{1}{\max}$. Then, each $\rho_i$ making up the type-0 run is called a type-1 run.

**Type-2 runs** Type-1 runs are further split into type-2 runs as follows. Let $\rho = s_0, (t_1, e_1), s_1, \ldots, (t_n, e_n), s_n$ be a type-1 run of $\mathcal{R}(\mathcal{H'})$, s.t. duration $(\rho) \leq T$. Let $S_\rho$ be the set of positions $0 < i \leq n$ s.t:

$$\exists x \in X : \left( \begin{array}{c} |\nu_{i-1}(x)| \neq |\nu_i(x)| \\ |\nu_{i-1}(x)| > 0 \text{ and } 0 = \langle \nu_{i-1}(x) \rangle < \langle \nu_i(x) \rangle \end{array} \right)$$

where $|x|$ and $\langle x \rangle$ denote respectively the integral and fractional parts of $x$. Roughly speaking, each transition $(t_i, e_i)$ with $i \in S_\rho$ corresponds to the fact that a variable changes its region, except in the case where the variable moves from $0^+$ to $(0, 1)$: such transitions are not recorded in $S_\rho$. Since $\rho$ is a type-1 run, its duration is at most $\frac{1}{\max}$. Hence, each variable can cross an integer value at most once along $\rho$, because all rates are positive. Thus, the size of $S_\rho$ can be bounded, by a value that does not depend on $|\rho|$.

**Lemma 4** Let $\rho$ be a type-1 run. Then $|S_\rho| \leq 3 \times |X|$.

**Proof.** As the duration of a type-1 run is $< \frac{1}{\max}$, each variable can, in the worst case, follow a trajectory that will be split into 4 parts. This happens when it starts in $(b, b+1)$, moves to $[b+1, b+1]$, then $(b+1, b+2)$, then gets reset and stays in $(0, 1)$.

Remark that if we had recorded in $S_\rho$ the indices of the transitions from $(\ell, \nu)$ to $(\ell', \nu')$ s.t. $\nu(x) = 0$ and $\nu(x) \in (0, 1)$ for some variable $x$, Lemma 4 would not hold, and we could not bound the size of $S_\rho$ by a value independent from $|\rho|$. Indeed, in any time interval, the density of time allows a variable to be reset and to reach a strictly positive value an arbitrary number of times.
Let us now split a type-1 run $\rho$ according to $S_\rho$. Assume $\rho = s_0, (t_1, e_1), s_1, \ldots, (t_n, e_n), s_n$, and that $S_\rho = \{p(1), \ldots, p(k)\}$, with $p(1) \leq p(2) \leq \cdots \leq p(k)$. Then, we let $\rho_0, \rho_1, \ldots, \rho_k$ be the runs s.t.:

$$\rho = \rho_0 \cdot s_{p(1)-1}, \ldots, \rho_{p(1)-1} \cdot s_{p(1)} \cdot \rho_1 \cdot s_{p(2)-1}, \ldots, \rho_{p(k)-1} \cdot s_{p(k)} \cdot \rho_k$$

(3)

Each $\rho_i$ is called a type-2 run, and can be empty. The next lemma summarises the properties of this construction:

**Lemma 5** Let $\rho$ be a type-1 run of $R(H')$ with duration $|\rho| \leq T$. Then, $\rho$ is split into:

$$\rho_0 \cdot \rho_1 \cdot \rho_2 \cdots \rho_k \cdot \rho_k$$

where each $\rho_i$ is a type-2 run; $k \leq 3 \times |X|$, $|\rho_i| = 1$ for all $1 \leq i \leq k$, and for all $1 \leq i \leq k$: $\rho_i = (t_0, e_0), (t_1, e_1), \ldots, (t_n, e_n)$, $(t_n, e_n)$ implies that, for all $x \in X$:

- either there is $a \in \mathbb{N}^{\geq 0}$ s.t. for all $0 \leq j \leq n$: $\nu_j(x) = a$ and $x$ is not reset along $\rho_i$;
- or for all $0 \leq j \leq n$: $\nu_j(x) \in (a, a + 1)$ with $a \in \mathbb{N}^{\geq 0}$ and $x$ is not reset along $\rho_i$;
- or for all $0 \leq j \leq n$: $\nu_j(x) \in [0, 1)$.

Remark that in the last case (i.e., $x$ is in $[0, 1)$ along a type-2 run), the number of resets cannot be bounded a priori. For the sake of clarity, we summarise the construction so far by the following lemma:

**Lemma 6** Each type-0 run of $R(H')$ can be decomposed into $k$ type-2 runs with $k \leq 3 \times (T \times \max + 1) \times |X|$.

**Type-3 runs** Finally, we obtain type-3 runs by splitting type-2 runs according to the first and last resets (if they exist) of each clock. Formally, let $s_0, (t_1, e_1), s_1, \ldots, (t_n, e_n), s_n$ be a type-2 run. Assume $Y_i$ is the reset set of $e_i$, for all $1 \leq i \leq n$. We let $FR_\rho = \{i \mid x \in Y_i \text{ and } \forall 0 \leq j < i: x \not\in Y_j\}$ and $LR_\rho = \{i \mid x \in Y_i \text{ and } \forall i < j \leq n: x \not\in Y_j\}$ be respectively the set of edge indices where a variable is reset for the first (last) in $\rho$. Let $R_\rho = FR_\rho \cup LR_\rho$ and assume $R_\rho = \{p(1), p(2), \ldots, p(k)\}$ with $p(1) \leq p(2) \leq \cdots \leq p(k)$. Then, we let $\rho_0, \rho_1, \ldots, \rho_k$ be the type 3 runs making up $\rho$ s.t. $\rho = \rho_0 \cdot s_{p(1)-1}, (t_{p(1)}, e_{p(1)}), s_{p(1)} \cdot \rho_1 \cdots s_{p(k)-1}, (t_{p(k)}, e_{p(k)}), s_{p(k)} \cdot \rho_k$.

Remark that each type-2 is split into at most $2 \times |X| + 1$ type-3 runs (i.e., $k \leq 2 \times |X|$).

**Contraction operator** So far, we have defined a procedure that splits any time-bounded run of $R(H')$ into a bounded number of type-3 runs. However, the construction does not allow us to bound the length of type-3 runs, because the density of time allows to perform an arbitrary number of actions in every possible time delay. Let us now define a contraction operator that turns type-3 runs into runs with the same effect but whose lengths can be uniformly bounded (thanks to the properties of type-3 runs established below).

Intuitively, the contraction operator works as follows. Let $\rho = (t_0, e_0), (t_1, e_1), (t_1, e_1), \ldots, (t_n, e_n), (t_n, e_n)$ be a run, and let $\pi$ be its timed path. We contract $\pi$ by looking for a pair of positions $i < j$ s.t. $\ell_i = \ell_j$ (i.e., $\pi[i + 1 : j]$ forms a loop) and s.t. all locations $\ell_{i+1}, \ell_{i+2}, \ldots, \ell_j$ occur in the prefix $\pi[1 : \ell_i]$. This situation is depicted in
variable valuations in the locations. This information will be particularly critical when the contraction operator relies on the fact that we have encoded the regions of the processes and that reaches the same state as the original run. Remark that the proof of soundness of \( \text{Cntr}^* (\pi) = \ell_0 + (t_6 + e_1, \ell_1, t_2, e_2, t_3, e_3, t_4, e_4, t_5, e_5, t_6, e_6, t_7, e_7, t_8, e_8, \ell_8) \)

Lemma 7 For all timed path \( \pi: |\text{Cntr}^* (\pi)| \leq |\text{Loc}|^2 + 1. \)

\begin{proof}
Assume \( \pi' = \text{Cntr}^* (\pi) = (t_1, e_1, \ldots, (t_n, e_n) \). Let \( \ell_0 = \text{src} (e_1) \), and \( \ell_i = \text{trg} (e_i) \) for all \( 1 \leq i \leq n \). Let \( \text{Loc}' = \{ L_0, \ldots, L_m \} \subseteq \text{Loc} \) be the set of locations that appear in \( \pi' \). For all \( L_i \in \text{Loc}' \), let \( k_i \) denote the least index s.t. \( \ell_{k_i} = L_i \) (i.e., the first occurrence of \( L_i \) in \( \pi' \)). Wlog,
we assume that \(k_0 \leq k_1 \leq \cdots \leq k_m\). Then, clearly, \(k_0 = 0\). Observe that each portion of the form \(\pi'[k_i : k_i + 1 - 1]\) (with \(0 \leq i \leq m - 1\)) is of length at most \(|\text{Loc}'|\).

Otherwise, the contraction operation can be applied in this portion, as there must be two positions \(k_i \leq \alpha < \beta \leq k_i + 1 - 1\) s.t. \(\ell_\alpha = \ell_\beta\), and all the locations occurring along \(\pi'[\alpha : \beta - 1]\) have occurred before, by definition of \(k_i\) and \(k_i + 1\). By the same arguments, \(|\pi'[k_m : n - 1]| \leq |\text{Loc}'|\) (remark that, by definition of the contraction operator, the last location \(\ell_n\) will never be considered for contraction). As \(\pi'[0 : n - 1]\) is made up of all those portions, and as there are \(|\text{Loc}'|\) portions, \(|\pi'|\) is bounded by \(|\text{Loc}'|^2 + 1 \leq |\text{Loc}|^2 + 1\). \(\Box\)

We can now lift the definition of the contraction operator to runs of type-2. Let \(\rho\) be a type-2 run and let us consider its (unique) decomposition into type-3 runs, as in \(3\), above. Then, we let \(\text{Cnt} (\rho) = \text{Run} (\text{first} (\rho), \pi_{\text{Cnt} (\rho)})\), where:

\[
\pi_{\text{Cnt} (\rho)} = \text{Cnt}^* (\text{TPath} (\rho_0)) \cdot (t_{p(1)}, e_{p(1)}), \text{Cnt}^* (\text{TPath} (\rho_1)) \cdot (t_{p(2)}, e_{p(2)}), \ldots, (t_{p(k)}, e_{p(k)}), \text{Cnt}^* (\text{TPath} (\rho_k))
\]

By definition of \(\text{Cnt}^*\), and by definition of \(\text{Cnt}\) on type-2 runs, it is easy to see that:

Lemma 8 For all type-3 runs \(\rho\): \(\text{duration} (\text{Cnt}^* (\text{TPath} (\rho))) = \text{duration} (\rho)\) and for all variables \(x\): \(\text{Effect} (\text{Cnt}^* (\text{TPath} (\rho))) (x) = \text{Effect} (\text{TPath} (\rho)) (x)\). Similarly, for all type-2 runs \(\rho\): \(\text{duration} (\pi_{\text{Cnt} (\rho)}) = \text{duration} (\rho)\) and for all variables \(x\): \(\text{Effect} (\pi_{\text{Cnt} (\rho)}) (x) = \text{Effect} (\text{TPath} (\rho)) (x)\).

Let us show that the contraction of type-2 runs is sound:

Proposition 1 For all type-2 runs \(\rho\), \(\text{Cnt} (\rho) \neq \bot\), first \(\text{Cnt} (\rho)\) = first \(\rho\) and last \(\text{Cnt} (\rho)\) = last \(\rho\).

Proof. Let \(\rho = (t_0, \nu_0), (t_1, e_1), \ldots, (t_n, \nu_n)\). Let \(\pi\) denote \(\text{TPath} (\rho)\), and let \(\pi_{\text{Cnt} (\rho)} = (t'_1, e'_1), \ldots, (t'_k, e'_k)\). For all \(1 \leq i \leq k\), let \(t'_i = \text{dest} (e'_i)\) by \(t'_i\); and let \(t'_0 = \text{src} (e'_1) = t_0\).

First, observe that, by definition of the contraction operator, \(\ell_0 = \ell'_1\). Let us show that \(\text{Cnt} (\rho) \neq \bot\). Assume that, for all \(0 \leq i \leq k\): \(t'_i = (\ell_{i, r_{i}})\) and let \(\nu'_i\) be the valuation s.t. for all \(x\):

\[
\nu'_i (x) = \begin{cases} 
\nu'_{i-1} (x) + \text{Rates} (t'_{i-1}) (x) \times t'_i & \text{If } e'_i \text{ does not reset } x \\
0 & \text{Otherwise}
\end{cases}
\]

Finally, let \(\nu'_0 = \nu_0\). Remark that \(\nu'_0 (x) \leq \nu'_1 (x) \leq \cdots \leq \nu'_k (x)\) because rates are non-negative. Clearly, to show that \(\rho' \neq \bot\), it is sufficient to show, for all \(i\), that \(\nu'_i \models g'_i\) (where \(g'_i\) is the guard of \(e'_i\); and that both \(\nu_i\) and \(\nu'_i\) satisfy\(\Box\) inv \(\ell_i\). For the sake of clarity, we prove that all the guards are satisfied; the arguments can be easily adapted to show that the invariants are satisfied too.

First, consider a variable \(x\) that is not reset along \(\pi\) (hence along \(\pi_{\text{Cnt} (\rho)}\) and s.t. \(\nu_0 (x) = \nu'_0 (x) > 0\). By definition of type-2 runs, and since \(x\) is not reset and not null initially, \(\nu_0 (x), \nu_1 (x), \ldots, \nu_n (x)\) all belong to the same interval \(I\) which is either \((a - 1, a)\) or \([a, a]\) for some \(a \geq 1\). Thus, in particular, \(\nu_0 (x) = \nu'_0 (x) \in I\). Moreover, since \(\text{Effect} (\pi_{\text{Cnt} (\rho)}) (x) = \text{Effect} (\pi) (x)\) (Lemma 8), we have \(\nu'_k (x) = \nu_n (x) \in I\) too. Hence, since \(\nu'_0 (x) \leq \nu'_1 (x) \leq \cdots \leq \nu'_k (x)\), we conclude that \(\nu'_i (x) \in I\) for all

\(^1\)Remember that we consider RHA\(^{\geq 0}\), so the invariants are convex.
0 ≤ i ≤ k. Since all the νi(x) are also in I, since ρ is a genuine run, and since all edges e′i in π∗ are also present in π, we conclude that ν ∈ I implies ν = ν′, for all valuation ν and all guards g′i of some edge e′i in π. Hence, ν′ = ν′, for all i.

Thus, we can, from now on, safely ignore all variables x that are not reset along π (hence along πCnt(ρ)) and s.t. ν0(x) = ν′0(x) > 0, and focus on variables x that are either reset along π or s.t. ν0(x) = ν′0(x) = 0. By definition of type-2 runs, in both cases, these variables take values in [0, 1) in each state along ρ. Hence, since ρ is region consistent (Lemma 3), all locations in ρ are of the form (t, r) with r(x) ∈ {0−, 0+, (0, 1)}, and so are all locations in πCnt(ρ): for all 0 ≤ i ≤ k: ℓi ∈ {0−, 0+, (0, 1)}. Let us denote, by ρ′j the value Run {first (ρ), πCnt(ρ) [1 : j]} for all m ≥ 1. We further denote by ρ′0 the run of null length (ℓ′0, ν0′). Let us show that, for all 0 ≤ j ≤ k, ρ′j ≠ ⊥, by induction on j.

The base case is j = 0 and is trivial since (ℓ0, ν0) = (ℓ′0, ν0′). For the inductive case, we assume that ρ′m−1 ≠ ⊥ (for some m ≥ 1) and ends in (ℓ, r), ν, and we show that we can extend it by firing (ℓ′m, ϵm) (i.e., that ρm ≠ ⊥). Observe that, by definition of Cnt, the edge ϵm occurs in πCnt(ρ) because it was already present in π (say, at position α, hence ϵα = ϵm and (ℓ, r) = ℓα−1). Moreover, still by definition of Cnt, the delay ℓ′m is equal to ℓα + ∑i=1β τp(i), where for all 1 ≤ i ≤ β: src (e p(i)) = (ℓ, r).

We consider three cases:

1. Either r(x) = 0−. In this case, since ρ and ρ′m−1 are region consistent (Lemma 3), and since the region r(x) is 0− (and not 0+), we know that ν0−1(x) = 0 (x is null when entering (ℓ, r) at position α − 1 in ρ), that ν(x) = 0 (x is null at the end of ρ′m−1), and that ν0−1(x) + τα × Rates(ℓ, r)(x) = τα × Rates(ℓ, r)(x) = 0 (x is null when leaving (ℓ, r) at position α − 1 in ρ). This means, in particular that it is sufficient for x to be null to satisfy the guard of ϵ′m = ϵα. Moreover, for all 1 ≤ i ≤ β: νp(i−1)(x) = τp(i) × Rates(ℓ, r)(x) (x is null when entering and leaving the locations at all positions p(i) that have yielded the contraction in ρ).

Thus, the value that x takes after letting ℓ′m t.u. elapse the last state or ρ′m−1 is ν′(x) = ν(x) + τm × Rates(ℓ, r)(x) = (ℓα + ∑i=1β τp(i)) × Rates(ℓ, r)(x) = 0. Hence ν′(x) satisfies the guard of ϵ′m, and we can extend ρ′m−1 by (ℓ′m, ϵ′m).

2. Or r(x) = 0+. In this case, we know that ν0−1(x) = ν(x) = 0, that τα × Rates(ℓ, r)(x) > 0, and that for all 1 ≤ i ≤ β: νp(i−1)(x) = 0 and τp(i) × Rates(ℓ, r)(x) > 0. Moreover, since duration (ρ) < 0−1, we can precisely this information and conclude that τα × Rates(ℓ, r)(x) ∈ (0, 1) and that for all 1 ≤ i ≤ β: τp(i) × Rates(ℓ, r)(x) ∈ (0, 1). Thus, it is sufficient, to satisfy the constraints on x in the guard of ϵ′m, that x ∈ (0, 1). Let us show that ν′(x) = (ℓα + ∑i=1β τp(i)) × Rates(ℓ, r)(x) is in (0, 1) too. We have ν′(x) > 0 because τα × Rates(ℓ, r)(x) > 0, as shown above. Moreover, ν′(x) < 1 because ℓα + ∑i=1β τp(i) ≤ duration (ρ) < 0−1, by def. of type-2 runs. Thus, ν′(x) satisfies the guard of ϵ′m and we can extend ρ′m−1 by (ℓ′m, ϵ′m).

3. Or r(x) = (0, 1). In this case, we can rely on the same arguments as above to show that ν′(x) > 0, and that ν′(x) should be in (0, 1) to satisfy the guard of ϵ′m. The difference with the previous case is that ν(x) ≠ 0 here, and we have to make sure that the additional delay accumulated on (ℓ, r) by the contraction operator does not increase x above 1. This property holds because of the split of type-2 runs in type-3 runs, according to the first reset of each variable. More precisely, we consider two cases. Either ℓα−1 occurs, in ρ in a type-3 run that
takes place after the first reset of \( x \). In this case, \( \nu'(x) = \nu(x)(t_\alpha + \sum_{i=1}^\beta t_i p(i)) \times \) Rates(\( \overline{t}, r \))(\( x \)) < 1, because all the \( t_i p(i) \) also occur \( \pi[\alpha : n] \) (i.e., after the first reset of \( x \)), and duration (\( \pi[\alpha : n] \)) < \( 1/\max \). Or \( \ell_{\alpha - 1} \) occurs, in \( \rho \) in a type-3 run that takes place before the first reset of \( x \). In this case, \( \nu'(x) = \nu(x)(t_\alpha + \sum_{i=1}^\beta t_i p(i)) \times \) Rates(\( \overline{t}, r \))(\( x \)) ≥ 1 implies that, in \( \rho \): \( \nu(p(i)) \geq 1 \), which contradicts the definition of type-2 runs. Hence, \( \nu'(x) \in (0, 1) \) and we can extend \( \rho'_{m - 1} \) by \( (t'_m, e'_m) \).

Let us conclude the proof by showing that \( \nu'_k = \nu_n \). We consider three cases. First, \( x \) is a variable that is not reset along \( \rho \). Since \( \text{Effect}(\text{Cnt}^*(\pi))(x) = \text{Effect}(\pi)(x) \) (Lemma 8), and since \( \nu_0 = \nu'_0 \), we conclude that \( \nu'_k(x) = \nu_n(x) \). Second, \( x \) is a variable that is reset along \( \rho \). Since the duration of a type-2 run is at most \( 1/\max \), \( \nu_n(x) \in [0, 1) \). Thus, we consider two further cases. Either \( \nu_n(x) = 0 \). Since \( \rho \) is region-consistent (Lemma 5), \( \ell_n \) is of the form \( (\overline{t}, r) \) with \( r(x) \in \{0^+, 0^\} \). However, \( \ell_n = \ell'_k \), and since Cnt(\( \rho \)) is a run and hence region-consistent, we conclude that \( \nu'_k(x) = 0 \). Or \( \nu_n(x) \in (0, 1) \). In this case, it is easy to observe that \( \nu_n(x) \) depends only on the portion of \( \rho \) that occurs after the last reset of \( x \), i.e., \( \nu_n(x) = \text{Effect}(\pi[i + 1 : n])(x) \), where \( i \) is the largest position in \( \rho \) s.t. \( e_i \) resets \( x \). By definition of the contraction operator on type 2 runs, \( e_i \) occurs at some position \( \alpha \) of \( \pi_{\text{Cnt}(\rho)} \), i.e. \( e_i = e'_\alpha \) and \( e'_\alpha \) is the last edge of \( \pi_{\text{Cnt}(\rho)} \) to reset \( x \). Thus, \( \nu'_k(x) = \text{Effect}(\pi_{\text{Cnt}(\rho)}[\alpha + 1 : k])(x) \). However, by Lemma 8 and by definition of the contraction of type 2 runs: \( \text{Effect}(\pi_{\text{Cnt}(\rho)}[\alpha + 1 : k])(x) = \text{Effect}(\pi[i + 1 : n])(x) \).

Then, observe that, by the above definition, and by Lemma 7 we can bound the length of Cnt(\( \rho \)) for type-2 runs \( \rho \):

**Lemma 9** For all type-2 runs: \( |\text{Cnt}(\rho)| \leq 8 \times |\text{Loc}|^2 \times |X| \).

**Proof.** By definition of type-2 runs, and by Lemma 7 \( |\text{Cnt}(\rho)| \) is at most \( 2 \times |X| + 1 \) \times \( |\text{Loc}|^2 + 1 \) \times \( 2 \times |X| + 1 \). Thus, \( |\text{Loc}| \geq 1 \) and \( |X| \geq 1 \). Hence \( |X| + 1 \leq 2 \times |X|, |\text{Loc}|^2 + 1 \leq 2 \times |\text{Loc}|^2 \). Hence the lemma.

We can now explain more intuitively why we need two different regions (\( 0^+ \) and \( 0^\)) for variables that are null, and cannot use \( [0, 0] \) instead. Consider the example given in Fig. 3 Run \( \rho_1 \) depicts a run of an automaton with a single variable \( x \), where we have used only region \( [0, 0] \) in the construction of \( R(\mathcal{H}') \). In this run, \( x \) is null in all four states. The two locations of \( R(\mathcal{H}') \) that are met are \( (t_1, [0, 0]) \) and \( (t_2, [0, 0]) \) (and in both locations, the rate of \( x \) is strictly positive). Hence, the contraction operator ‘merges’ the two occurrences of both locations, an produces \( \rho_2 \). However, \( \rho_2 \) fails to satisfy Proposition 1 as \( x \) is null in the last state of \( \rho_1 \) but not in the last state of \( \rho_2 \). This comes from the fact that region \( [0, 0] \) does not allow to distinguish between locations that are left with a strictly positive delay or a null delay. With our definition of \( R(\mathcal{H}') \), however, the first state of the run is \( (t_1, 0^\), \( 0 \), \( x \) is null when crossing the first edge, but the third state is \( ((t_1, 0^+), 0) \), as \( x \) is not null when crossing the last edge, which avoids the problem illustrated in Fig. 3.

Thus, summing up the properties of the contraction operator, and the splitting procedure we obtain, as a corollary of Proposition 1 and Lemma 6:

**Corollary 1** Let \( s \) and \( s' \) be two states of \( R(\mathcal{H}') \). Then, \( R(\mathcal{H}') \) admits a \( T \) time-bounded type-0 run \( \rho \) with first (\( \rho \)) = \( s \) and last (\( \rho \)) = \( s' \) iff it admits a \( T \) time bounded type-0 run \( \rho' \) with first (\( \rho' \)) = \( s \), last (\( \rho' \)) = \( s' \) and \( |\rho'| \leq 48 \times T \times \max \times |\text{Loc}|^2 \times |X| \) of
Before contraction: \( \rho_1 = ((\ell_1, [0, 0]), 0) \rightarrow ((\ell_2, [0, 0]), 0) \)

When we cross this edge, \( x \) is null.

After contraction: \( \rho_2 = ((\ell_1, [0, 0]), 0) \rightarrow ((\ell_2, [0, 0]), t_1 + t_2) \)

We reach a state where \( x \) is not null anymore.

Figure 3: An example that shows why the contraction operator fails if we use \([0, 0]\) to characterise the variables that are null.
Finally, for all SHA \( \mathcal{H} \geq 0 \) \( \mathcal{H} = (X, \text{Loc}, \text{Edges}, \text{Rates}, \text{Inv}, \text{Init}) \) and all time bound \( T \in \mathbb{N} \), we let:

\[
F(\mathcal{H}, T) = 24 \times (T \times r_{\text{max}} + 1) \times |X|^2 \times |\text{Loc}|^2 \times (2 \times c_{\text{max}} + 1)^{2 \times |X|}
\]

This value \( F(\mathcal{H}, T) \) is actually a bound on the length of the runs we need to consider to decide \( T \)-time-bounded reachability:

**Theorem 1** Let \( \mathcal{H} \) be a SHA \( \geq 0 \), \( T \) be a time bound and let \( s_1 \) and \( s_2 \) be two states of \( \mathcal{H} \). Then \( \mathcal{H} \) admits a \( T \)-time-bounded run \( \rho \) with first \( (\rho) = s_1 \) and last \( (\rho) = s_2 \) if it admits a \( T \)-time-bounded run \( \rho' \) with \( |\rho'| \leq F(\mathcal{H}, T) \), first \( (\rho') = s_1 \) and last \( (\rho') = s_2 \).

**Proof.** The if direction is trivial, let us prove the only if, by proving the contraposition, i.e., that if \( \mathcal{H} \) admits no \( T \)-time-bounded run of length at most \( F(\mathcal{H}, T) \) from \( s_1 \) to \( s_2 \), then it admits no \( T \)-time-bounded run from \( s_1 \) to \( s_2 \). By Lemma [1] if \( \mathcal{H} \) admits no \( T \)-time-bounded run of length at most \( F(\mathcal{H}, T) \) from \( s_1 = (\ell_1, \nu_1) \) to \( s_2 = (\ell_2, \nu_2) \), then, \( \mathcal{H}' \) admits no \( T \)-time-bounded run of length at most \( F(\mathcal{H}, T) \) from \( s_1' = ((\ell_1, r_1), \nu_1) \) to \( s_2' = ((\ell_2, r_2), \nu_2) \). By Corollary [2] and by [2], \( \mathcal{R}(\mathcal{H}) \) admits no \( T \)-time-bounded run from \( s_1 \) to \( s_2 \), regardless of the length of the run. Hence, by Lemma [2] \( \mathcal{H}' \) admits no \( T \)-time-bounded run \( \rho \) from \( s_1 \) to \( s_2 \), and neither does \( \mathcal{H} \), by Lemma [1] again.

Remark that \( F(\mathcal{H}, T) = \mathcal{O}(T \times 2^{|\mathcal{H}|}) \), where \( |\mathcal{H}| \) is the number of bits necessary to encode \( \mathcal{H} \), using standard encoding techniques and binary encoding for the constants. Hence, Theorem [1] tells us that, to decide \( T \)-time-bounded reachability, we only need to consider runs whose length is singly exponential in the size of the instance \( (\mathcal{H}, T) \).

Let us now briefly explain how we can adapt the previous construction to cope with non-singular rates. Let us first notice that given \( \mathcal{H} \) a RHA \( \geq 0 \), the construction of R(\mathcal{H'}\) still makes perfect sense and still satisfies Lemma [1]. Then, we need to adapt the definition of timed path. A timed path is now of the form \((t_1, R_1, e_1), \ldots, (t_n, R_n, e_n)\), where each \( R_i : X \rightarrow \mathbb{R} \) gives the actual rate that was chosen for each variable at the \( i \)-th continuous step. It is then straightforward to extend the definitions of Cnt, Effect and Contraction to take those rates into account and still keep the properties needed to prove Theorem [1]. More precisely, the contraction of a set of transitions \((t_1, R_1, e_1), \ldots, (t_n, R_n, e_n)\) yields a transition \((t, R, e)\) with \( t = \sum_{i=1}^{n} t_i \) and, \( R = \sum_{i=1}^{n} t_i \times R_i \). Note that we need to rely on the convexity of the invariants and rates in an RHA to ensure that this construction is correct. Thus, we can extend Theorem [1] to the case of RHA with positive rates (RHA\( \geq 0 \)).

**Corollary 2** Let \( \mathcal{H} \) be a RHA \( \geq 0 \), \( T \) be a time bound and let \( s_1 \) and \( s_2 \) be two states of \( \mathcal{H} \). Then \( \mathcal{H} \) admits a \( T \)-time-bounded run \( \rho \) with first \( (\rho) = s_1 \) and last \( (\rho) = s_2 \) if it admits a \( T \)-time-bounded run \( \rho' \) with \( |\rho'| \leq F(\mathcal{H}, T) \), first \( (\rho') = s_1 \) and last \( (\rho') = s_2 \).
4 Time-bounded reachability is \textsc{NEXPTIME}-c

In this section, we establish the exact computational complexity of the time-bounded reachability problem for \textsc{RHA}$_{\geq 0}$.

**Theorem 2** The time-bounded reachability problem for \textsc{RHA}$_{\geq 0}$ is complete for \textsc{NEXPTIME}.

To prove this theorem, we exhibit an \textsc{NEXPTIME} algorithm for time-bounded reachability and we reduce this problem from the reachability problem of exponential time Turing machine.

**An \textsc{NEXPTIME} algorithm** Recall that an instance of the time-bounded reachability problem is of the form \((\mathcal{H}, \ell, T)\), where \(\mathcal{H}\) is an \textsc{RHA}$_{\geq 0}$, \(\ell\) is a location, and \(T\) is a time bound (expressed in binary). We establish membership to \textsc{NEXPTIME} by giving a non-deterministic algorithm that runs in exponential time in the size of \((\mathcal{H}, \ell, T)\) in the worst case. The algorithm first guesses a sequence of edges \(E = e_0e_1 \ldots e_n\) of \(\mathcal{H}\) s.t. \(n + 1 \leq F(\mathcal{H}, T)\) and \(\text{trg}(e_n) = \ell\). Then the algorithm builds from \(E\) a linear constraint \(\Phi(E)\), that expresses all the properties that must be satisfied by a run that follows the sequence of edges in \(E\) (see [13] for a detailed explanation on how to build such a constraint). This constraint uses \(n + 1\) copies of the variables in \(X\) and \(n + 1\) variables \(t_i\) to model the time elapsing between two consecutive edges, and imposes that the valuations of the variables along the run are consistent with the rates, guards and resets of \(\mathcal{H}\). Finally, the algorithm checks whether \(\Phi(E)\) is satisfiable and returns ‘yes’ iff it is the case.

The number of computation steps necessary to build \(\Phi(E)\) is, in the worst case, exponential in \(|\mathcal{H}|\) and \(T\). Moreover, checking satisfiability of \(\Phi(E)\) can be done in polynomial time (in the size of the constraint) using classical algorithms to solve linear programs. Clearly this procedure is an \textsc{NEXPTIME} algorithm for solving the time-bounded reachability problem for \textsc{RHA}$_{\geq 0}$.

**\textsc{NEXPTIME}-hardness** To establish the \textsc{NEXPTIME}-hardness, we show how to reduce the membership problem for non-deterministic exponential time Turing machines to time-bounded reachability for \textsc{SHA}$_{\geq 0}$.

A non-deterministic exponential time Turing machine (NExpTM) is a tuple \(M = (Q, \Sigma, \Gamma, \xi, q_0, \delta, F, \xi)\) where \(Q\) is the (nonempty and finite) set of control states, \(\Sigma\) is the alphabet of the tape, \(\Gamma \supseteq \Sigma\) is the alphabet of the tape, \(\xi \in \Gamma\) is the blank symbol, \(q_0 \in Q\) is the initial control state, \(\delta \subseteq Q \times \Gamma \times \Gamma \times \{L, R\} \times Q\) is the transition relation, \(F \subseteq Q\) is the set of accepting states, and \(\xi = \mathcal{O}(2^{p(|n|)})\) (for some polynomial \(p\)), is an exponential function that bounds the execution time of the machine on input \(w\) by \(\xi(|w|)\).

As usual, a state of \(M\) is a triple \((q, w_1, w_2)\) where \(q \in Q\) is a control state, \(w_1 \in \Gamma^*\) a word that represents the content of the tape on the left of the reading head (this word is empty when the head is on the leftmost cell of the tape), and \(w_2 \in \Gamma^*\) is the content of the tape on the right of the reading head excluding the sequence of blank symbols (\(\xi\)) at the end of the tape, (in particular the first letter in \(w_2\) is the content of the cell below the reading head).

A transition of the Turing machine is a tuple of the form \((q_1, \gamma_1, \gamma_2, D, q_2)\) with the usual semantics: it is enabled if the current control state is \(q_1\), the content of the cell below the reading head is equal to \(\gamma_1\), and the head should not
be above the left most cell when \( D = L \). The execution of the transition modifies the content of the tape below the reading head to \( \gamma_2 \), moves the reading head one cell to the right if \( D = R \), or one cell to the left if \( D = L \), and finally, changes the control state to \( q_2 \). We write \((q, w_1, w_2) \mapsto (q', w_1', w_2')\) if there exists a transition in \( \delta \) from state \((q, w_1, w_2)\) to state \((q', w_1', w_2')\).

An (exponentially bounded) execution of \( M \) on input \( w \) is a finite sequence of states \( c_0 c_1 \ldots c_n \) such that: (i) \( n \leq \xi(|w|) \) (the execution is exponentially bounded); (ii) \( c_0 = (q_0, \epsilon, w, 2^{\xi(|w|)-|w|}) \), (the initial control state is \( q_0 \) and the tape contains \( w \) followed by the adequate number of blank symbols); and (iii) for all \( 0 \leq i < n \), \( c_i \mapsto c_{i+1} \), (the transition relation is enforced). The execution is accepting iff \( c_n = (q, w_1, w_2) \) with \( q \in F \). W.l.o.g., we make the assumption that \( \Sigma = \{0, 1\} \), \( \Gamma = \{0, 1, \#\} \), and transitions only write letters in \( \Sigma \). This ensures that in all reachable states \((q, w_1, w_2)\) we have that \( w_1, w_2 \in \{0, 1\}^* \).

The membership problem for an \( \text{NExpTM} \) \( M \) and a word \( w \) asks whether there exists an accepting execution of the Turing Machine \( M \) that uses at most \( \xi(|w|) \) steps.

Let us show how we can encode all executions of \( M \) into the executions of an

\( \text{SHA}^{\geq 0} \) \( H_M \). We encode the words \( w_1 \) and \( w_2 \) as pairs of rational values \((l_1, c_1)\) and \((l_2, c_2)\) where \( l_1 = \frac{1}{2^{\xi(|w|)-|w|}} \) encodes the length of the word \( w_1 \) by a rational number in \([0, 1]\), and \( c_1 \) encodes \( w_1 \) as follows. Assume \( w_1 = \sigma_0 \sigma_1 \ldots \sigma_n \). Then, we let 

\[ c_1 = \text{Val}_r(w_1) = \sigma_0 \cdot \frac{1}{2} + \sigma_1 \cdot \frac{1}{4} + \sigma_2 \cdot \frac{1}{8} + \cdots + \sigma_{n-1} \cdot \frac{1}{2^n} + \sigma_n \cdot \frac{1}{2^n}. \]

Intuitively, \( c_1 \) is the value which is represented in binary by \( 0.\sigma_0 \sigma_1 \ldots \sigma_{n-1} \sigma_n \), i.e., \( w_1 \) is the binary encoding of the fractional part of \( c_1 \) where the most significant bit in the rightmost position. For instance, if \( w_1 = 001010 \) then \( \text{Val}_r(w_1) = 0 \cdot \frac{1}{2} + 0 \cdot \frac{1}{4} + 1 \cdot \frac{1}{8} + 0 \cdot \frac{1}{16} + 1 \cdot \frac{1}{32} + 0 \cdot \frac{1}{64} = 0.3125 \), and so \( w_1 \) is encoded as the pair \((\frac{1}{64}, 0.3125)\). Remark that we need to remember the actual length of the word \( w_1 \) because the function \( \text{Val}_r(\cdot) \) ignores the leading 0’s (for instance, \( \text{Val}_r(\text{001010}) = \text{Val}_r(1010) \)). Symmetrically, if \( w_2 = \sigma_0 \sigma_1 \ldots \sigma_n \), we let \( c_2 = \text{Val}_r(w_2) = \sigma_0 \cdot \frac{1}{2} + \sigma_1 \cdot \frac{1}{4} + \cdots + \sigma_{n-1} \cdot \frac{1}{2^n} + \sigma_n \cdot \frac{1}{2^n} \) (i.e., \( \sigma_0 \) is now the most significant bit). Then, a state \((q, w_1, w_2)\) of the TM is encoded as follows: the control state \( q \) is remembered in the locations of the automaton, and the words \( w_1, w_2 \) are stored, using the encoding described above using four variables to store the values \((l_1, c_1)\) and \((l_2, c_2)\).

With this encoding in mind, let us list the operations that we must be able to perform to simulate the transitions of the TM. Assume \( w_1 = w_0^1 w_1^2 \ldots w_n^1 \) and \( w_2 = w_0^2 w_1^2 \ldots w_n^2 \). We first describe the operations that are necessary to read the tape:

- **Read the letter under the head.** Following our encoding, we need to test the value of the bit \( w_0^1 \). Clearly, \( w_0^1 = 1 \) iff \( l_2 \leq 1/2 \), and \( c_2 \geq \frac{1}{2} \), \( w_0^1 = 0 \) iff \( l_2 \leq 1/2 \), and \( c_2 < \frac{1}{2} \) and \( w_0^1 = 0 \) iff \( c_1 < \frac{1}{2} \).

- **Test whether the head is in the leftmost cell of the tape.** This happens if and only if \( w_1 = \epsilon \), and so if and only if \( l_1 = 1 \).

- **Read the letter at the left of the head (assuming that \( w_1 \neq \epsilon \)).** Following our encoding, this amounts to testing the value of the bit \( w_1^1 \). Clearly, \( w_1^1 = 1 \) iff \( c_1 \geq \frac{1}{2} \) and \( w_1^1 = 0 \) iff \( c_1 < \frac{1}{2} \).

Then, let us describe the operations that are necessary to update the values on the tape. Clearly, they can be carried out by appending and removing 0 or 1’s to the right of \( w_1 \) or to the left of \( w_2 \). Let us describe how we update \( c_1 \) and \( l_1 \) to simulate these operations on \( w_1 \) (the operations on \( w_2 \) can be deduced from this description). We denote by \( c_1' \) (resp. \( l_1' \)) the value of \( c_1 \) (\( l_1 \)) after the simulation of the TM transition.
In this section, we show that Corollary 2 implies that we can effectively compute the set of states that are reachable within $T$ time units in an RHA with non-negative rates (using formulas of the first-order logic $(\mathbb{R}, 0, 1, +, \leq)$ over the reals as a symbolic representation for such sets). We demonstrate, by means of two examples, that this information can be useful in practice, in particular when the regular (not time-bounded) fixed points do not terminate.

**Post and Pre** Let $s$ be state of an RHA with set of edges $\text{Edges}$. Then, we let $\text{Post}(s) = \{s' \mid \exists e \in \text{Edges}, t \in \mathbb{R}^+ : s \xrightarrow{t} e \rightarrow s'\}$ and $\text{Pre}(s) = \{s' \mid \exists e \in \text{Edges}, t \in \mathbb{R}^+ : s' \xleftarrow{t} e \rightarrow s\}$. We further let $\text{Reach}^{\leq T}(s) = \{s' \mid \exists \pi : s \xrightarrow{\pi} s' \land \text{duration} (\pi) \leq T\}$, and $\text{coReach}^{\leq T}(s) = \{s' \mid \exists \pi : s' \xleftarrow{\pi} s \land \text{duration} (\pi) \leq T\}$ be respectively the set of states that are reachable from $s$ (that can reach $s$) within $T$ time units. We extend all those operators to sets of states in the obvious way.

5 Computing fixpoints

In this section, we show that Corollary 2 implies that we can effectively compute the set of states that are reachable within $T$ time units in an RHA with non-negative rates (using formulas of the first-order logic $(\mathbb{R}, 0, 1, +, \leq)$ over the reals as a symbolic representation for such sets). We demonstrate, by means of two examples, that this information can be useful in practice, in particular when the regular (not time-bounded) fixed points do not terminate.

**Post and Pre** Let $s$ be state of an RHA with set of edges $\text{Edges}$. Then, we let $\text{Post}(s) = \{s' \mid \exists e \in \text{Edges}, t \in \mathbb{R}^+ : s \xrightarrow{t} e \rightarrow s'\}$ and $\text{Pre}(s) = \{s' \mid \exists e \in \text{Edges}, t \in \mathbb{R}^+ : s' \xleftarrow{t} e \rightarrow s\}$. We further let $\text{Reach}^{\leq T}(s) = \{s' \mid \exists \pi : s \xrightarrow{\pi} s' \land \text{duration} (\pi) \leq T\}$, and $\text{coReach}^{\leq T}(s) = \{s' \mid \exists \pi : s' \xleftarrow{\pi} s \land \text{duration} (\pi) \leq T\}$ be respectively the set of states that are reachable from $s$ (that can reach $s$) within $T$ time units. We extend all those operators to sets of states in the obvious way.
Figure 4: Gadgets (i) for multiplication by 2, (ii) adding \( \frac{1}{2} \) and (iii) subtracting \( \frac{1}{2} \). The rates of the \( y \not\in \{x, z\} \) is 0. Gadget (i) can be modified to divide by 2, by swapping the rates of \( x \) and \( z \) in the second location. \( x_0 \) is the value of \( x \) when entering the gadget.
Region algebra\footnote{The notion of region used in this section differs from the notion of region given by \textit{Reg} \((c_{\text{max}}, X)\) and used to define \(R(\mathcal{H})\). Notice however that any region from \(R(\mathcal{H})\) can be expressed via a quantifier free formula of \((\mathbb{R}, 0, 1, +, \leq)\) with free variables in \(X\). The converse is obviously not true.} To symbolically manipulate sets of states, it is well known that we can use formulas of \((\mathbb{R}, 0, 1, +, \leq)\), i.e. the first-order logic of the reals, with the constants 0 and 1, the usual order \(\leq\) and addition + (see \cite{11} for the details). Recall that the satisfiability problem for that logic is decidable \cite{4} and that it admits effective quantifier elimination. Further remark that, in a RHA, all guards can be characterised by a formula of \((\mathbb{R}, 0, 1, +, \leq)\) ranging over \(X\). Let \(\Psi\) be a formula of \((\mathbb{R}, 0, 1, +, \leq)\), and let \(\nu\) be a valuation of the free variables of \(\Psi\). Then, we write \(\nu \models \Psi\) iff \(\nu\) satisfies \(\Psi\), and we let \(\llbracket \Psi \rrbracket\) be the set off all valuations \(\nu\) s.t. \(\nu \models \Psi\). To emphasise the fact that a formula \(\Psi\) ranges over the set of variables \(X\), we sometimes denote it by \(\Psi(X)\).

Based on \((\mathbb{R}, 0, 1, +, \leq)\), we can define a so-called \textit{algebra of regions} \cite{11} to effectively represent sets of states. The region algebra in that algebra can be seen as functions \(R\) from the set of locations \(\text{Loc}\) to quantifier free formula of \((\mathbb{R}, 0, 1, +, \leq)\) with free variables in \(X\), representing sets of valuations for the variables of the RHA. More precisely, any region \(R\) represents the set of states \(\llbracket R \rrbracket = \{ (\ell,\nu) \mid \nu \in \llbracket R(\ell) \rrbracket \}\). As \((\mathbb{R}, 0, 1, +, \leq)\) is closed under all Boolean operations, so is the region algebra. Since the logic is decidable, testing whether \(s \in \llbracket R \rrbracket\) or whether \(\llbracket R \rrbracket = \emptyset\) are both decidable problems.

In order to obtain fixpoint expressions that characterise \(\text{Reach}^{\leq T}(s)\) and \(\text{coReach}^{\leq T}(s)\) using the region algebra, we introduce \(\text{post}^2\) and \(\text{pre}^2\) operators ranging over regions. Let \(R\) be a region. We let \(\text{post}^2(R)\), be the region s.t. for all \(\ell \in \text{Loc}\), \(\text{post}^2(R)(\ell)\) is obtained by eliminating quantifiers in \(\Psi^E(X) \lor \Psi^I(X)\), where \(\Psi^E(X)\) characterises all the successors of \(R(\ell)\) by an edge with source \(\ell\), and \(\Psi^I(X)\) represents all the successors of \(R(\ell)\) by a flow transition in \(\ell\) (time elapsing). The following equations define \(\Psi^E(X)\) and \(\Psi^I(X)\), both ranging on the set of free variables \(X\):

\[
\Psi^E_\ell = \bigvee_{e \in \text{Edges}} \psi^e_\ell
\]

\[
\psi^e_{\ell, \ell', x, x'}(\ell, x, x') = \exists X' : \begin{cases}
R(\ell)(X') \land g(X') \\
\land \bigwedge_{x \in X' \setminus Y} x = x' \land \bigwedge_{x \in Y} x = 0 \\
\land \bigwedge_{x \in X} \text{Inv}(\ell)(X') \land \text{Inv}(\ell')(X)
\end{cases}
\]

\[
\Psi^I_\ell = \exists t : \exists X' : \begin{cases}
t \geq 0 \land R(\ell)(X') \\
\land \text{Inv}(\ell)(X) \land \text{Inv}(\ell')(X') \\
\land \bigwedge_{x \in X} x' + t \cdot \min(\text{Rates}(\ell, x)) \leq x \\
\land \bigwedge_{x \in X} x \leq x' + t \cdot \max(\text{Rates}(\ell, x))
\end{cases}
\]
predecessors of $R(\ell)$ by a flow transition in $\ell$:

$$
\Phi_{\ell}^E = \bigvee_{e \in \text{Edges}} \varphi_{\ell}^e
$$

$$
\varphi_{\ell}^{(e, g, Y, \ell')} = \exists X' : \begin{cases}
R(\ell')(X') \land g(X) \\
\land \bigwedge_{x \in X \land Y} x = x' \land \bigwedge_{x \in Y} x' = 0
\end{cases}
$$

$$
\Phi_{\ell} = \exists t : \exists X' : \begin{cases}
t \geq 0 \land R(\ell)(X') \\
\land \text{Inv}(\ell)(X) \land \text{Inv}(\ell')(X') \\
\land \bigwedge_{x \in X} x + t \cdot \min(\text{Rates}(\ell, x)) \leq x'
\end{cases}
$$

To keep the above definitions compact, we have implicitly assumed that the rates are given as closed rectangles. The definitions of $\Phi_{\ell}^E$ and $\Psi_{\ell}^E$ can be adapted to cope with intervals that are left (respectively right) open by substituting $<$ ($>$) for $\leq$ ($\geq$).

In practice formulas in $(\mathbb{R}, 0, 1, +, \leq)$ can be represented and manipulated as finite union of convex polyhedra for which there exist efficient implementations, see [3] for example. Those techniques have been implemented in HYTECH [8] and PHAVER [6]. Unfortunately, termination of the symbolic model-checking algorithms is not ensured for linear hybrid automata. While in the literature, it is known that forward reachability and backward reachability fixpoint algorithms terminate for initialised rectangular hybrid automata [10], we show here that termination is also guaranteed for time-bounded fixpoint formulas over the class of RHA$^{\geq 0}$ (that are not necessarily initialised).

**Time-bounded forward and backward fixpoints** Let $\mathcal{H}$ be an RHA$^{\geq 0}$ with set of variables $X$, and let $T \in \mathbb{N}$ be a time bound. Let us augment $\mathcal{H}$ with a fresh variable $t$ to measure time (hence the rate of $t$ is 1 in all locations, and $t$ is never reset). Let $R$ be region over the variables $X$. Then, it is easy to see that the following fixpoint equations characterise respectively $\text{Reach}^{\leq T}([R])$ and $\text{coReach}^{\leq T}([R])$:

$$
\text{Reach}^{\leq T}([R]) = \mu Y \cdot (([R(X)] \cup \text{Post}(Y)) \cap [0 \leq t \leq T])
$$

$$
\text{coReach}^{\leq T}([R]) = \mu Y \cdot (([R(X)] \cup \text{Pre}(Y)) \cap [0 \leq t \leq T])
$$

The next lemma ensures that these fixpoints can be effectively computed. The proof rely on Corollary [2].

**Lemma 10** For all RHA$^{\geq 0}$ $\mathcal{H}$, all region $R$ and all time bound $T$, the least fix points (4) and (5) are respectively equal to the limit of $F_0, F_1, F_2, \ldots$ and $B_0, B_1, B_2, \ldots$ where:

$$
F_0 = [R(X) \land 1 \leq t \leq T] \\
F_i = (\text{Post}(F_{i-1}) \cap [0 \leq t \leq T]) \cup F_{i-1} \quad \text{for all } i > 0 \\
B_0 = [R(X) \land 1 \leq t \leq T] \\
B_i = B_i = (\text{Pre}(B_{i-1}) \cap [0 \leq t \leq T]) \cup B_{i-1} \quad \text{for all } i > 0
$$

Furthermore, both sequences stabilize after at most $F(\mathcal{H}, T)$ iterations, and both fixpoints can be computed in worst-case doubly exponential time.
Proof. We justify the result for the least fixpoint equation (4), the result for the least fixpoint equation (5) is justified similarly.

By induction, it is easy to prove that, for all $i \geq 0$, $F_i$ contains all the states that are reachable within $T$ time units and by at most $i$ transitions. By Corollary, we know that all states that reachable within $T$ time units are reachable by a run of length at most $F(H, T)$. We conclude that $F_j = F_{j+1} = \text{Reach}^{\leq T}([R])$ for $j = F(H, T)$. All the operations for computing $F_i$ from $F_{i-1}$ take polynomial time in the size of $F_{i-1}$, and so the size of $F_i$ is also guaranteed to be polynomial in $F_{i-1}$, the overall doubly-exponential time bound follows.

Note that by our NExPTIME-hardness result, this deterministic algorithm can be considered optimal (unless NExPTIME=ExPTIME). Let us now consider two examples to demonstrate that this approach can be applied in practice.

**Example 1: Leaking gas burner**

We present an example of a system where the classical fixpoint computation for reachability analysis does not terminate, while the time-bounded analysis does terminate. Consider the example of a leaking gas burner [1]. The gas burner can be either leaking or not leaking. Leakages are repaired within 1 second, and no leakage can happen in the next 30 seconds after a repair. In Fig. 5 (top), an automaton with two locations and the clock $x$ is a model of the gas burner. In order to measure the leakage time and the total elapsed time, the stopwatch $t$ and clock $y$ are used as monitors of the system. It was shown using backward reachability analysis that in any time interval of at least 60 seconds, the time of leakage is at most one twentieth of the elapsed time [8]. The fixpoint is computed after 7 iterations of the backward reachability algorithm. However, the forward reachability analysis does not terminate.

Using forward time-bounded reachability analysis we can prove the property that in all time intervals of fixed length $T \geq 60$, the leakage time is at most $\frac{T}{20}$. In order to prove that this property holds in all time intervals, we perform the reachability analysis from all possible states of the system (i.e., from location leaking with $0 \leq x \leq 1$, and location not_leaking with $x \geq 0$) and starting with $t = y = 0$. For a fixed time bound $T$, we compute the set of reachable states satisfying $y \leq T$ and check that $t \leq \frac{T}{20}$ when $y = T$. The results of this paper guarantees that the analysis terminates. Using

\[
\begin{align*}
0 \leq x \leq 1 & \quad x := 0 \quad \text{not leaking} \\
\dot{x} &= 1 \\
\dot{y} &= 1 \\
\ell_0 &= 0 \\
\ell_1 &= \dot{x} = 3 \\
\dot{y} &= 2 \\
\ell_2 &= y = 1 \\
y := 0 \\
\ell_3 &= \dot{x} = 1 \\
\dot{y} &= 1 \\
\ell_4 &= \dot{t} = 0 \\
x = 0 \\
x = y = 0 \\
0 \leq x,y \leq 1 & \quad x := 0 \\
\dot{x} &= 1 \\
\dot{y} &= 1 \\
\ell_0 &= 0 \\
\ell_1 &= \dot{x} = 3 \\
\dot{y} &= 2 \\
\ell_2 &= y = 1 \\
y := 0 \\
\ell_3 &= \dot{x} = 1 \\
\dot{y} &= 1 \\
\ell_4 &= \dot{t} = 0 \\
\ell_5 &= t \geq 30 \\
x \geq 0 \\
x \geq 30 & \quad x := 0 \\
0 \leq x \leq 1 & \quad \text{leaking} \\
\dot{x} &= 1 \\
\dot{y} &= 1 \\
\ell_0 &= 0 \\
\ell_1 &= \dot{x} = 3 \\
\dot{y} &= 2 \\
\ell_2 &= y = 1 \\
y := 0 \\
\ell_3 &= \dot{x} = 1 \\
\dot{y} &= 1 \\
\ell_4 &= \dot{t} = 0 \\
x \geq 0 \\
x \geq 30 & \quad x := 0
\end{align*}
\]

Figure 5: A stopwatch automaton for the leaking gas burner (top) and an SHA with bounded invariants (bottom).
HyTech, the property is established for $T = 60$ after 5 iterations of the forward time-bounded fixpoint algorithm. Thus for all time intervals of $T = 60$ seconds, the leakage time is at most $T/20$.

**Example 2: bounded invariant** In Fig. 5 (bottom), we consider a rectangular automaton with positive rates where all variables have a bounded invariant $[0, 1]$. In this example, the forward reachability analysis of HyTech does not terminate because the set of reachable states is not a finite union of polyhedra (see Fig. 5). On the other hand, the time-bounded forward fixpoint terminates by Lemma 10. This example shows that it is not sufficient to bound the variables in the automaton to get termination, but it is necessary to bound the time horizon of the analysis.

Figure 6: Reachable states for the automaton of Fig. 5 (bottom).

**References**

[1] R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In *Hybrid Systems*, LNCS 736, pages 209–229. Springer, 1993.

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

[3] R. Bagnara, P. M. Hill, and E. Zaffanella. The parma polyhedra library: Toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems. *Sci. Comput. Program.*, 72(1-2):3–21, 2008.

[4] S. Basu. New results on quantifier elimination over real closed fields and applications to constraint databases. *J. ACM*, 46(4):537–555, 1999.

[5] T. Brihaye, L. Doyen, G. Geeraerts, J. Ouaknine, J.-F. Raskin, and J. Worrell. On reachability for hybrid automata over bounded time. In *ICALP (2)*, volume 6756 of *Lecture Notes in Computer Science*, pages 416–427. Springer, 2011.

[6] G. Frehse. Phaver: algorithmic verification of hybrid systems past hytech. *STTT*, 10(3):263–279, 2008.

[7] T. A. Henzinger. The theory of hybrid automata. In *LICS*, pages 278–292. IEEE Computer Society, 1996.

[8] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. A user guide to HyTECH. In *TACAS’95: Tools and Algorithms for the Construction and Analysis of Systems*, volume 1019 of *Lecture Notes in Computer Science*, pages 41–71. Springer-Verlag, 1995.
We prove by (backward) induction (on $\ell$) that when $\ell_n = 0$, there is nothing to prove. Let us now assume that given $\rho = ((\ell_0, \nu_0), (\ell_1, \nu_1), \ldots, (\ell_n, \nu_n))$ of $H'$, we associate a run $\rho' = ((\ell_0, \nu_0), (t_0, e_0'), ((\ell_1, r_1), \nu_1), \ldots, (t_n, e_n'), ((\ell_n, \nu_n))$ of $R(H')$ (hence with duration $\rho = \left(\sum_{i=0}^{n} r_i(x)\right)$ s.t.:

- for all $0 \leq i \leq n$, for all $x \in X$ (assuming $t_{n+1} = 0$):
  \[ r_i(x) = \begin{cases} 
    0^+ & \text{if } \nu_i(x) = 0 \text{ and } (t_{i+1} > 0 \text{ and } \dot{x} \neq 0) \\
    0^- & \text{if } \nu_i(x) = 0 \text{ and } (t_{i+1} = 0 \text{ or } \dot{x} = 0) \\
    \nu_i(x) & \text{otherwise}
  \end{cases} \]
- $e_i'$ is the unique edge between $(\ell_i, r_i)$ and $(\ell_{i+1}, r_{i+1})$, corresponding to $e_i$.

We prove by (backward) induction (on $n$) that the run $\rho'$ is a genuine run of $R(H')$. When $n = 0$, there is nothing to prove. Let us now assume that given $\rho = ((\ell_0, \nu_0), (t_0, e_0'), (\ell_1, \nu_1), \ldots, (t_n, e_n'))$ of $H'$, we have proved that $((\ell_0, r_0), \nu_0), (t_0, e_0'), ((\ell_1, r_1), \nu_1)$ is a genuine run of $R(H')$. To obtain the desired result, it remains to prove that $((\ell_0, r_0), \nu_0), (t_0, e_0'), ((\ell_1, r_1), \nu_1)$ is a genuine run of $R(H')$. For this, we have to prove that for all $x \in X$: (i) $\nu_0(x) + t \times \text{Rates}(\ell_0, r_0)(x) \models \text{Inv}(\ell_0, r_0)$, for all $0 \leq t \leq t_1$, (ii) $\nu_0(x) + t_1 \times \text{Rates}(\ell_0, r_0)(x) \models g'$ (where $g'$ is the guard of the transition $e_1'$), and (iii) $\nu_1(x) = 0$ (resp. $\nu_1(x) = \nu_0(x)$ if $x \in Y'$ (resp. $x \notin Y'$)) (where $Y'$ is the reset of the transition $e_1'$). Let us distinguish three cases:

1. Case 1: $r(x) = 0^+$. In this case, by construction of $\rho'$, we know that $\nu_0(x) = 0$, $t_1 > 0$ and $\dot{x} \neq 0$. In particular, we have that $\nu_0(x) + t_1 \times \text{Rates}(\ell_0, r_0)(x) > 0$.

By construction of $R(H')$, we know that $g'(x) = g(x) \land r'(x) \land (x > 0)$, where
$g(x)$ (resp. $g'(x)$) represents the constraints on $x$ in the guard of the transition $e_1$ (resp. $e'_1$), and $r \leq_{t_5} Rates(\ell_0, r_0)$ $\rho''$. Moreover we have that $Inv(\ell_0, r_0)(x) = Inv(\ell_0)(x)$. Since $\rho$ is a genuine run of $\mathcal{H}'$, we clearly have that (i) and (iii) are satisfied. Point (ii) follows from the facts that $\rho$ is a genuine run of $\mathcal{H}'$ and that $\nu_0(x) + t_1 \times Rates(\ell_0, r_0)(x) > 0$.

2. Case 2: $r(x) = 0^\omega$. In this case, by construction of $\rho'$, we know that $\nu_0(x) = 0$, $t_1 = 0$ or $\hat{x} = 0$. In particular, we have that $\nu_0(x) + t_1 \times Rates(\ell_0, r_0)(x) = 0$. By construction of $R(\mathcal{H}')$, we know that $g'(x) = g(x) \land r''(x) \land (x = 0)$, where $g(x)$ (resp. $g'(x)$) represents the constraints on $x$ in the guard of the transition $e_1$ (resp. $e'_1$), and $r \leq_{t_5} Rates(\ell_0, r_0)$ $\rho''$. Moreover we have that $Inv(\ell_0, r_0)(x) = Inv(\ell_0)(x) \land (x = 0)$. Since $\rho$ is a genuine run of $\mathcal{H}'$, we clearly have that (iii) is satisfied. Points (i) and (ii) follow from the facts that $\rho$ is a genuine run of $\mathcal{H}'$ and that $\nu_0(x) + t \times Rates(\ell_0, r_0)(x) = 0$, for all $0 \leq t \leq t_1$.

3. Case 3: $r(x) \not\in \{0^\omega, 0^+\}$. This case is simpler than the two previous ones. The three points (i), (ii) and (iii) follow from the facts that $\rho$ is a genuine run of $\mathcal{H}'$.

Then, with each run $\rho'$, we associate the run $\rho = ((\ell_0, r_0), (t_0, e_0', \nu_0), ((\ell_1, r_1), (t_1, e_1), \nu_1), \ldots, (t_n, e_n', \nu_n))$ of $R(\mathcal{H}')$ where, for all $1 \leq i \leq n$, $e_i$ is the unique edge of $\mathcal{H}'$ that corresponds to $e'_i$. Since the guards and invariant of $R(\mathcal{H}')$ are more constraining than those of $\mathcal{H}'$, the fact that $\rho'$ is a genuine run of $R(\mathcal{H}')$ implies that $\rho$ is a genuine run of $\mathcal{H}'$. □

Proof of Lemma

Let us first prove that, for all $0 \leq i \leq n$: $\nu_i \in r_i$. The proof is by induction on $i$. For $i = 0$, the property holds by definition of $R(\mathcal{H}')$ and because $((\ell_0, r_0), \nu_0)$ is an initial state. Assume $\nu_i \in r_i$ for some $i \geq 0$, and let us show that $\nu_{i+1} \in r_{i+1}$. Let $g$ and $Y$ denote respectively the guard and the reset set of $e_{i+1}$. Let $\nu' = \nu_i + t_{i+1} \times Rates(\ell_i)$ be the valuation of the variables when crossing $e_{i+1}$. By construction, we know that there is a region $r''$ which is a conjunct of $g$ (hence $\nu' \in r''$) s.t. for all $x \not\in Y$: $r''(x) = r_{i+1}(x)$. Thus, for all $x \not\in Y$: $\nu'(x) = \nu_{i+1}(x) \in r_{i+1}(x)$. Moreover, still by construction, for all $x \in Y$: $r_{i+1}(x) \in \{0^\omega, 0^+\}$. Hence, for all $x \in Y$: $\nu_{i+1}(x) = 0 \in r_{i+1}$ too.

To conclude, the two last points of the lemma follow immediately from the construction of $R(\mathcal{H}')$, as for all edge $e$ and all variable $x$ s.t. $src(e) = (l, r)$ and $r(x) = 0^\omega$ (resp. $r(x) = 0^+$), the constraint $x = 0$ ($x > 0$) appears as a conjunct of $e$’s guard. □

\[\text{Notice that it makes sense to decouple guard according to variables since there are no diagonal constraints.}\]