Time-Bounded Reachability for Monotonic Hybrid Automata: Complexity and Fixed Points[⋆]

T. Brihaye[a], L. Doyen[b], G. Geeraerts[c], J. Ouaknine[d], J.F. Raskin[c], and J. Worrell[d]

[a] UMons, Belgium; [b] LSV Cachan, France; [c] ULB, Belgium; [d] Oxford University, UK

Abstract. We study the *time-bounded reachability problem* for *monotonic hybrid automata* (MHA), i.e., rectangular hybrid automata for which the rate of each variable is either always non-negative or always non-positive. In this paper, we revisit the decidability results presented in [5] and show that the problem is NExpTime-complete. We also show that we can effectively compute fixed points that characterise the sets of states that are reachable (resp. co-reachable) within $\mathbf{T}$ time units from a given 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 interacts with a continuous environment. The need for modeling and analysing hybrid systems is thus obvious.

Hybrid automata are arguably among the most prominent families of models for hybrid systems [7]. Hybrid automata are finite automata (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). The variables evolve with time elapsing, at a rate which is given by a flow that depends on the current location of the automaton. The theory of hybrid automata has been well developed for about two decades, and tools to analyse them are readily available, for instance HyTech [8, 9].

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 [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 [12, 14]. For instance, while *language inclusion* is, in general undecidable for timed automata, it is decidable when considering only executions of *bounded duration* [14]. Following this line of research, we have recently investigated the decidability of *time-bounded reachability* for rectangular hybrid automata, i.e., whether a given state is reachable by an execution of duration at most $\mathbf{T}$, for a given $\mathbf{T}$ [5]. We have shown

⋆ This work has been partly supported by a grant from the National Bank of Belgium, the ARC project (number AUWB-2010-10/15-UMONS-3), the FRFC project (number 2.4545.11) and a ‘Crédit aux chercheurs’ of the FRS – F.N.R.S.

that *time-bounded* reachability is *decidable* for *rectangular hybrid automata with non-negative rates* (RHA$^{\geq 0}$), while it is well-known that (plain, time unbounded) reachability is undecidable for this class [10]. We have also shown that the decidability frontier is quite sharp: time-bounded reachability becomes *undecidable* once we allow either diagonal constraints in the guards or a single variable to have both positive and negative rates. The decidability result relies on a so-called *contraction operator* that allows to construct, from any run of duration at most $\mathbf{T}$ of an RHA$^{\geq 0}$ $\mathcal{H}$, 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 the automaton $\mathcal{H}$ and the bound $\mathbf{T}$. Hence, deciding reachability within $\mathbf{T}$ time units reduces to exploring runs of bounded lengths only, which is algorithmically feasible [5]. Yet, this yields only a *non-deterministic algorithm with doubly exponential time complexity* for a strict subclass of RHA$^{\geq 0}$, and no lower bound is given.

In the present work, we revisit and extend the results from [5], both from the theoretical and the practical points of view. *First*, we consider the class of *monotonic hybrid automata* (MHA for short) which are rectangular hybrid automata where the rate of each variable is either always non-negative or always non-positive (thus, MHA generalise RHA$^{\geq 0}$). *Second*, we provide a *new contraction operator* that allows to derive a *singly exponential upper bound* on the lengths of the runs that need to be considered, thereby providing an NEXPTIME algorithm for the whole class of MHA. *Third*, 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*. *Fourth*, we extend those results towards practical applications, by showing that we *can effectively compute* the set of states that are reachable (resp. co-reachable) within $\mathbf{T}$ time units, from a given state. Finally, 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 manage to compute, using HYTECH, the set of states reachable within $\mathbf{T}$ time units for values of $\mathbf{T}$ that are sufficient to prove non-trivial properties of those examples

Note that the missing proofs can be found in the companion technical report [6].

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 $\dot{X} = \{\dot{x} \mid x \in X\}$ be the set of dotted variables, corresponding to the variables' time 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*, and over $\dot{X}$ 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 $\mathcal{H} = (X, \mathrm{Loc}, \mathrm{Edges}, \mathrm{Rates}, \mathrm{Inv}, \mathrm{Init})$ where $X = \{x_1, \ldots, x_{|X|}\}$ is a finite set of continuous *variables* ; Loc is a finite set of *locations*; $\mathrm{Edges} \subseteq \mathrm{Loc} \times \mathcal{G}(X) \times$

$2^X \times \mathrm{Loc}$ is a finite set of *edges*; $\mathrm{Rates} : \mathrm{Loc} \mapsto \mathcal{R}(X)$ assigns to each location a constraint on the *possible variable rates*; $\mathrm{Inv} : \mathrm{Loc} \mapsto \mathcal{G}(X)$ assigns an *invariant* to each location; and $\mathrm{Init} \subseteq \mathrm{Loc}$ is a *set of initial locations*. For an edge $e = (\ell, g, Y, \ell')$, we denote by $\mathsf{src}(e)$ and $\mathsf{trg}(e)$ the locations ℓ and ℓ' respectively, g is called the *guard* of e and Y is the *reset* set of e. In the sequel, we denote by rmax and cmax the maximal constants occurring respectively in the constraints of $\{\mathrm{Rates}(\ell) \mid \ell \in \mathrm{Loc}\}$ and of $\{\mathrm{Rates}(\ell) \mid \ell \in \mathrm{Loc}\} \cup \{g \mid \exists(\ell, g, Y, \ell') \in \mathrm{Edges}\}$.

An LHA has *non-negative rates* if for all variables x, for all locations ℓ, the constraint $\mathrm{Rates}(\ell)$ 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 $\mathrm{Rates} : \mathrm{Loc} \times X \to \mathcal{I}$ that associate with each location ℓ and each variable x an interval of possible rates $\mathrm{Rates}(\ell)(x)$. A *monotonic hybrid automaton* (MHA) is an RHA such that, for all variable x: either $\mathrm{Rates}(\ell, x) \subseteq [0, +\infty)$ in all locations ℓ; or $\mathrm{Rates}(\ell, x) \subseteq (-\infty, 0]$ in all locations ℓ. A *singular hybrid automaton* (SHA) is an RHA such that for all locations ℓ and for all variables x: $\mathrm{Rates}(\ell)(x)$ is a singleton. We note SMHA and $\mathrm{RHA}^{\geq 0}$ for *singular* MHA, *non-negative rates* RHA resp. Note that MHA generalises $\mathrm{RHA}^{\geq 0}$.

LHA semantics A *valuation* of a set of variables X is a function $\nu : X \mapsto \mathbb{R}$. We denote by $\mathbf{0}$ the valuation that assigns 0 to each variable. For a valuation x of X and a guard $g \in \mathcal{G}(X)$, we write $v \models g$ iff v *satisfies* g. Given an LHA $\mathcal{H} = (X, \mathrm{Loc}, \mathrm{Edges}, \mathrm{Rates}, \mathrm{Inv}, \mathrm{Init}, X)$, a *state* of $\mathcal{H}$ is a pair (ℓ, ν), where $\ell \in \mathrm{Loc}$ and ν is a valuation of X. The semantics of $\mathcal{H}$ is defined as follows. For a state $s = (\ell, \nu)$ of $\mathcal{H}$, an *edge step* $(\ell, \nu) \xrightarrow{e} (\ell', \nu')$ can occur and change the state to (ℓ', ν') if $e = (\ell, g, Y, \ell') \in \mathrm{Edges}$, $\nu \models g$, $\nu'(x) = \nu(x)$ for all $x \notin Y$, and $\nu'(x) = 0$ for all $x \in Y$; for a time delay $t \in \mathbb{R}^+$, a *continuous time step* $(\ell, \nu) \xrightarrow{t} (\ell, \nu')$ can occur and change the state to (ℓ, ν') if there is a vector $r = (r_1, \ldots r_{|X|})$ such that $r \models \mathrm{Rates}(\ell)$, $\nu' = \nu + (r \cdot t)$, and $\nu + (r \cdot t') \models \mathrm{Inv}(\ell)$ 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 $\mathsf{trg}(e_i) = \mathsf{src}(e_{i+1})$ 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, ℓ, we denote by $\pi[k : \ell]$ the maximal portion $(t_i, e_i), (t_{i+1}, e_{i+1}), \ldots, (t_j, e_j)$ of π such that $\{i, i+1, \ldots, j\} \subseteq [k, \ell]$ (note that the interval $[k, \ell]$ could be empty, in which case $\pi[k : \ell]$ would be 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 $\mathsf{Effect}(\pi) = \sum_{i=1}^{n} \mathrm{Rates}(\ell_{i-1}) \cdot t_i$ be the *effect of* π (where $\ell_i = \mathsf{src}(e_i)$ for all $1 \leq i \leq n$).

A *run* in $\mathcal{H}$ is a sequence $s_0, (t_1, e_1), s_1, (t_2, e_2), \ldots, (t_n, e_n), s_n$ s.t. (i) $(t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$ is a timed path in $\mathcal{H}$, and (ii) for all $0 \leq i < n$, there exists a state s'_i of $\mathcal{H}$ with $s_i \xrightarrow{t_{i+1}} s'_i \xrightarrow{e_{i+1}} s_{i+1}$. Given a run $\rho = s_0, (t_1, e_1), \ldots, s_n$, let $\mathsf{first}(\rho) = s_0$, $\mathsf{last}(\rho) = s_n$, $\mathsf{duration}(\rho) = \sum_{i=1}^{n} t_i$, and $|\rho| = n + 1$. We say that ρ is $\mathbf{T}$*-time-bounded* (for $\mathbf{T} \in \mathbb{N}$) if $\mathsf{duration}(\rho) \leq \mathbf{T}$. 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 $s_n = s'_0$, 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 $\mathsf{TPath}(\rho) = (t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$, 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

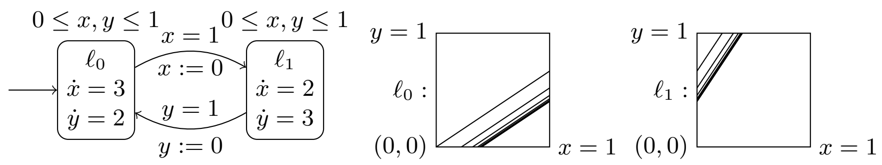

Fig. 1. An MHA with its set of reachable states.

notation and denote a run ρ with $\mathsf{first}(\rho) = s_0$, $\mathsf{last}(\rho) = s$ and $\mathsf{TPath}(\rho) = \pi$ by $s_0 \xrightarrow{\pi} s$. The converse however is not true: given a timed path π and an initial state s_0, it could be impossible to build a run starting from s_0 and following π because some guards or invariants along π might be violated. However, *when the automaton is singular*, such a run is necessarily unique if it exists, and we denote by $\mathsf{Run}(s_0, \pi)$ the function that returns the unique run ρ such that $\mathsf{first}(\rho) = s_0$ and $\mathsf{TPath}(\rho) = \pi$ if it exists, and $\perp$ otherwise. Note that, when considering 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 ρ: $\nu_n(x) = \nu_0(x) + \mathsf{Effect}(\pi)(x)$.

Time-bounded reachability problem While the reachability problem asks whether there is a run reaching a given goal location, we consider only runs with *bounded duration*.

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

This problem is decidable [5] for RHA$^{\ge 0}$, but its exact complexity was left open until now. We prove in Section 4 that it is **NExpTime-complete for MHA**. This problem is known to become undecidable either when diagonal constraints are allowed in the guards, or when non-monotonic RHA are considered [5]. In Section 5, we extend these results by showing how to compute a finite and algorithmically manipulable representation of the set of states that are reachable within $\mathbf{T}$ time units.

Let us illustrate, by means of the MHA (actually an RHA$^{\ge 0}$) $\mathcal{H}$ in Fig. 1 (left), the difficulties encountered when computing the reachable states. In this example, one can show that the set of reachable states is not a finite union of polyhedra, see Fig. 1 (right). Moreover, one can observe that the number of bits necessary to encode the states reachable from $(\ell_0, 0, 0)$ grows *linearly* with the length of the run. This example shows that finding an adequate, compact and effective representation (such as regions in the case of Timed Automata [2]) for the set of reachable states of an MHA is not trivial (and, in full generality, impossible because reachability is undecidable for this class). Nevertheless, in Section 5, we show that, for MHA, an effective representation of the set of states that are reachable *within* $\mathbf{T}$ *time units* can be computed.

3 Bounding the length of time-bounded runs

In this section, we prove the main technical result of the paper. For the sake of clarity, we consider a *singular* MHA $\mathcal{H} = (X, \mathrm{Loc}, \mathrm{Edges}, \mathrm{Rates}, \mathrm{Inv}, \mathrm{Init})$ and explain later why the results extend to general MHA. The result we prove is that '$\mathcal{H}$ *can reach a state* s *within* $\mathbf{T}$ *time unit iff it can reach* s *within* $\mathbf{T}$ *time unit by a run of bounded length*, where the bound is *uniform*: it depends only on $\mathbf{T}$ and on the number $|\mathcal{H}|$ of bits necessary to encode $\mathcal{H}$ (with standard encoding for the constants). More precisely, let $F(\mathcal{H}, \mathbf{T}) = 24 \times (\mathbf{T} \times \text{rmax} + 1) \times |X|^2 \times |\text{Loc}|^2 \times (2 \times \text{cmax} + 3)^{2 \times |X|}$. Then:

Theorem 2. *Let* $\mathcal{H}$ *be an SMHA,* $\mathbf{T}$ *be a time bound and let* s_1 *and* s_2 *be two states of* $\mathcal{H}$*. Then* $\mathcal{H}$ *admits a* $\mathbf{T}$*-time-bounded run* ρ *with* $\mathsf{first}(\rho) = s_1$ *and* $\mathsf{last}(\rho) = s_2$ *iff it admits a* $\mathbf{T}$*-time-bounded run* ρ' *s.t.* $|\rho'| \leq F(\mathcal{H}, \mathbf{T})$*,* $\mathsf{first}(\rho') = s_1$ *and* $\mathsf{last}(\rho') = s_2$*.*

This theorem will be used in the next sections to obtain optimal algorithms for deciding time-bounded reachability. Observe that $F(\mathcal{H}, \mathbf{T}) = \mathcal{O}\left(\mathbf{T} \times 2^{|\mathcal{H}|}\right)$. Thus, Theorem 2 says that, to decide $\mathbf{T}$-time-bounded reachability, we only need to consider runs whose length is exponential in the size of the instance $(\mathcal{H}, \mathbf{T})$.

We establish this result in two steps. First, we show that **each time-bounded run can be split into a *bounded* number of so-called type-2 (sub-)runs** (see hereunder for the definitions of type-0, type-1 and type-2 runs). Because of the density of time, we cannot bound the length of those type-2 runs, yet we show that they enjoy properties that allow us to **replace each type-2 run by an equivalent run of bounded length**. By *equivalent* we mean a run that starts and ends in the same states, and has the same duration. Combining the bounds on the number of type-2 runs and on the length of the runs we substitute to the original type-2 runs, we obtain Theorem 2.

Contraction operator To obtain the bounded length runs that we substitute to the original type-2 runs, we rely on a *contraction operator*. As this operator is central to our proof we start by describing it intuitively[1]. Let $\rho = (\ell_0, \nu_0), (t_1, e_1), (\ell_1, \nu_1), \ldots, (t_n, e_n), (\ell_n, \nu_n)$ be a run, and let $\pi = \mathsf{TPath}(\rho)$. We *contract* π 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 $\pi[1 : i]$. An example is the timed path of the run ρ in the top of Fig. 2. Then, the contraction consists, roughly speaking, in *deleting* the portion $\pi[i+1 : j]$ from π, and in *reporting* the delays $t_{i+1}, \ldots, t_{j-1}$ to the other occurrences of $\ell_i, \ldots, \ell_{j-1}$ in π (that exist by hypothesis), see Fig. 2. Obviously, this contraction returns a timed path *with shorter length*. We show (Lemma 7 hereunder) that, by repeatedly applying this contraction, we obtain a timed path $\mathsf{Cnt}^*(\pi)$ whose length is bounded by $|\text{Loc}|^2 + 1$, i.e. a value that does not depend on the length of π.

Now, we can lift the definition of the contraction operator to runs: for a run ρ, $\mathsf{Cnt}(\rho) = \mathsf{Run}(\mathsf{first}(\rho), \mathsf{Cnt}^*(\mathsf{TPath}(\rho)))$. Clearly, there is, in general, no guarantee that this contracted run exists, i.e. that $\mathsf{Cnt}(\rho) \neq \bot$ (see examples hereunder). However, we will show that, when correctly applied to so-called type-2 runs (see hereunder for the precise definition), $\mathsf{Cnt}(\rho)$ produces a genuine run of bounded length that starts and ends in the same states as the original run.

Let us now discuss several concrete examples of this contraction procedure. In all these examples, we assume an MHA with a single variable x whose rate is 1 in all locations, and we consider the run ρ depicted in Fig. 2. We also let $\pi' = \mathsf{Cnt}(\mathsf{TPath}(\rho))$

[1] The definition of this operator is crucial to obtain the NEXPTIME algorithm in Section 4. It differs from the one introduced in [5], which does not allow to obtain an NEXPTIME algorithm.

$$\begin{tikzcd}
\rho = (\ell_0, \nu_0) \arrow[r, "{t_1, e_1}"] & (\ell_1, \nu_1) \arrow[r, "{t_2, e_2}"] \arrow[rr, no head, bend right, "\text{cycle}"'] & (\ell_0, \nu_2) \arrow[r, "{t_3, e_3}"] \arrow[ll, dashed, bend right] & (\ell_1, \nu_3) \arrow[r, "{t_4, e_4}"] \arrow[ll, dashed, bend right] & (\ell_2, \nu_4)
\end{tikzcd}$$

$$\begin{tikzcd}
\mathsf{Cnt}\left(\mathsf{TPath}\left(\rho\right)\right) = \ell_0 \arrow[r, "{t_1 + t_3, e_1}"] & \ell_1 \arrow[r, "{t_2 + t_4, e_4}"] & \ell_2
\end{tikzcd}$$

Fig. 2. Illustrating the contraction operator

and $\rho' = \mathsf{Run}\left((\ell_0, \nu_0), \pi'\right)$ – thus, ρ' could be equal to $\perp$. *First* assume that $\nu_0(x) = 0$, that $t_1 = t_2 = t_3 = t_4 = .1$ and that all edges $e_1, \ldots, e_4$ dot not reset x. In this case, ρ' is a genuine run that reaches $(\ell_2, .4)$. However, as remarked above, there are many cases where either $\rho' = \perp$ or $\rho' \neq \perp$ but does not reach the same state as ρ. Let us observe four of these cases, as they will be used later to justify our constructions.
Case 1. Assume x is never reset along ρ, $\nu_0(x) = 0$, $t_1 = t_3 = 1$ and the guard of e_1 is $x = 1$. Then, $\rho' = \perp$ as $\nu_0(x) + t_1 + t_3 = 2$ and does not satisfy the guard of e_1. Intuitively, the problem occurs because x crosses value 1 along ρ, and the compression reports the delay t_3, occurring *after* x crosses 1 in the original run, to a part where $x \leq 1$ in the original run. To avoid this, we split the run once a variable changes its *region*, where the regions are $[0, 1)$ and all $[a, a]$, $(a, a + 1)$ for $a \geq 1$. Since we consider time-bounded runs, we obtain a *bounded number of sub-runs*. Note that we *do not* split when a variable moves from $[0, 0]$ to $(0, 1)$ or vice-versa, because the density of time allows a variable to be reset and to increase strictly an *unbounded* number of times in any time interval. Hence, this splitting strategy is not sufficient to guarantee that $\mathsf{Cnt}\left(\rho\right) \neq \perp$ and is equivalent to ρ, as shown by the next three cases, where x is in $[0, 1)$ along ρ.
Case 2. Assume $\rho' \neq \perp$, e_1 resets x, e_2, e_3 and e_4 do not reset x and $t_1 = t_2 = t_3 = t_4 = .1$. Then, $\nu_4(x) = t_2 + t_3 + t_4 = .3$. Observe that $\nu_4(x)$ depends only on the run portion that occurs *after* e_1 because e_1 is the last edge to reset x. On the other hand, ρ' reaches a state (ℓ_2, ν) with $\nu(x) = t_2 + t_4 = .2 \neq \nu_4(x)$, because the contraction *reports, before the last reset (e_1), the delay t_3 that occurs after the last reset in ρ.*
Case 3. Assume $\nu_0(x) = .8$, $t_1 = t_3 = .1$, the guard of e_1 is $x < 1$ and e_1 resets x. Then, $\rho' = \perp$, as $\nu_0(x) + t_1 + t_3 = 1$, which does not satisfy the guard of e_1. Intuitively, the problem occurs because the time delay t_3 that takes place *after the first reset of x* in ρ has been reported *before the first reset of x*.
Case 4. Assume $\nu_0(x) = 0$, $t_1 = 0$, $t_2 = t_3 = t_4 = .1$, e_2, e_3 and e_4 reset x, and the guard of e_1 is $x = 0$. Further assume that that x has just been reset when entering ℓ_0. Then, $\rho' = \perp$, as $\nu_0(x) + t_1 + t_3 = .1$, which does not satisfy the guard of e_1. Intuitively, the problem occurs because, x is null when entering *and* leaving the first occurrence of ℓ_0, while it is null when entering and non-null when leaving the second occurrence of ℓ_0. Thus, the time delay t_3 should not be reported to the first occurrence of ℓ_0. To avoid this, we *label locations* with special *regions* telling us whether x is null when leaving the location (region $\mathbf{0}^=$) or not ($\mathbf{0}^+$), and we will forbid the contraction operator to report delay from one location to another with different regions.

The actual splitting into type-2 runs proceeds stepwise: we split a run into type-1 runs, then each type-1 in type-2 runs, so that we avoid the pitfalls described above. As explained in the discussion of case 1 above, we first need to track the *regions* of the variables, thanks to the following construction.

Region labelling Let $\mathsf{Reg}\,(\mathrm{cmax}) = \big(\{[a,a],\ (a-1,a) \mid a \in \{1,\ldots,\mathrm{cmax}\}\} \cup \{\mathbf{0}^{=}, \mathbf{0}^{+}, (\mathrm{cmax},+\infty)\}\big)$ be the set of *regions*, and further let $\mathsf{Reg}\,(\mathrm{cmax}, X)$ denote the set of all functions $r : X \mapsto \mathsf{Reg}\,(\mathrm{cmax})$ that assign a region to each variable. By abuse of language, we sometimes call *regions* elements of $\mathsf{Reg}\,(\mathrm{cmax}, X)$ too. Remark that the definition of $\mathsf{Reg}\,(\mathrm{cmax}, X)$ differs from the classical regions [2] by the absence of $[0,0]$ which is replaced by two symbols: $\mathbf{0}^{=}$ and $\mathbf{0}^{+}$, and by the fact that no information is retained about the relative values of the fractional parts of the variables. The reason of the introduction of the two regions $\mathbf{0}^{=}$ and $\mathbf{0}^{+}$ is to avoid the problem occurring in Case 4 above. When testing for membership to a region, $\mathbf{0}^{+}$ and $\mathbf{0}^{=}$ should be interpreted as $[0,0]$, i.e., $v \in \mathbf{0}^{+}$ and $v \in \mathbf{0}^{=}$ hold iff $v = 0$. Given a valuation ν of the set of variable X, and $r \in \mathsf{Reg}\,(\mathrm{cmax}, X)$, we let $\nu \in r$ iff $\nu(x) \in r(x)$ for all x, and, provided that $\nu > \mathbf{0}$, we denote by $[\nu]$ the (unique) element from $\mathsf{Reg}\,(\mathrm{cmax}, X)$ s.t. $\nu \in [\nu]$. Remark that for all sets of variable X and all maximal constants cmax: $|\mathsf{Reg}\,(\mathrm{cmax}, X)| \leq (2\times\mathrm{cmax}+3)^{|X|}$. Let r_1 and r_2 be two regions in $\mathsf{Reg}\,(\mathrm{cmax}, X)$, 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 $r_1 \leq^{v}_{\mathrm{ts}} r_2$) 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 $r_1 \leq^{v}_{\mathrm{ts}} r_2$, $r_1(x) = \mathbf{0}^{=}$ and $r_2(x) = \mathbf{0}^{+}$ for some variable x (for instance, if $v(x) = 0$).

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

- $\mathrm{Loc}' = \mathrm{Loc} \times \mathsf{Reg}\,(\mathrm{cmax}, X)$ and $\mathrm{Init}' = \mathrm{Init} \times \{\mathbf{0}^{=}, \mathbf{0}^{+}\}^{X}$
- for all $(\ell, r) \in \mathrm{Loc}'$: $\mathrm{Rates}'(\ell, r) = \mathrm{Rates}(\ell)$; $\mathrm{Inv}(\ell, r) = \mathrm{Inv}(\ell) \wedge \bigwedge_{x:r(x)=\mathbf{0}^{=}} x = 0$
- There is an edge $e' = \big((\ell, r), g \wedge x \in r'' \wedge g_0, Y, (\ell', r')\big)$ in Edges' iff there are an edge $e = (\ell, g, Y, \ell')$ in Edges and a region r'' s.t.: $r \leq^{\mathrm{Rates}(\ell)}_{\mathrm{ts}} r''$, for all $x \notin Y$: $r'(x) = r''(x)$, for all $x \in Y$: $r'(x) \in \{\mathbf{0}^{=}, \mathbf{0}^{+}\}$ and $g_0 = \bigwedge_{x \in X} g_0(x)$ where for all $x \in X$: $g_0(x) = (x = 0)$ if $r(x) = \mathbf{0}^{=}$; $g_0(x) = (x > 0)$ if $r(x) = \mathbf{0}^{+}$; and $g_0(x) = \mathbf{true}$ otherwise. 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 (ℓ, r) and (ℓ', r').

It is easy to see that this construction incurs an exponential blow up in the number of locations, but preserves reachability of states. More precisely, $|\mathrm{Loc}'| \leq |\mathrm{Loc}| \times |\mathsf{Reg}\,(\mathrm{cmax}, X)| = |\mathrm{Loc}| \times (2 \times \mathrm{cmax} + 3)^{|X|}$ and:

Lemma 3. *$\mathcal{H}$ admits a run ρ with* $\mathsf{first}\,(\rho) = (\ell, \nu)$ *and* $\mathsf{last}\,(\rho) = (\ell', \nu')$ *iff there are r and r' s.t.* $\mathsf{R}\,(\mathcal{H})$ *admits a run ρ' with* $\mathsf{first}\,(\rho') = ((\ell, r), \nu)$, $\mathsf{last}\,(\rho') = ((\ell', r'), \nu')$, $\mathsf{duration}\,(\rho) = \mathsf{duration}\,(\rho')$ *and* $|\rho| = |\rho'|$.

Intuitively, the regions that label locations in $\mathsf{R}\,(\mathcal{H})$ 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, to avoid the issue with the contraction operator described in case 4 above. This explains the two regions, $\mathbf{0}^=$ and $\mathbf{0}^+$, corresponding to 0. They encode the fact that the variable is null (resp. strictly positive) when leaving the location.

Type-0 and type-1 runs Without loss of generality, we assume that, if a state is reachable, then it is reachable by a run of the same duration which can be split into at most $\mathbf{T} \times \mathrm{rmax} + 1$ portions of duration $< \frac{1}{\mathrm{rmax}}$. In practice, this can be achieved by adding one self-loop on all locations of $\mathsf{R}(\mathcal{H})$, which does not impact time-bounded reachability. Such runs of ρ of $\mathsf{R}(\mathcal{H})$ are called *type-0 runs* and are of the form $\rho = \rho_0 \cdot \rho_1 \cdots \rho_k$ s.t. for all $0 \leq i \leq k$: $\mathsf{duration}(\rho_i) < \frac{1}{\mathrm{rmax}}$. Each ρ_i is called a *type-1 run. Intuitively*, each variable will *cross at most one integer value different from 0* in each type 1 run, because the automaton is *monotonic*. For instance, if $x \in (2, 3)$ at the beginning of a type-1 run, then x can reach $(3, 4)$ along the run, but will never cross 4. However, x could be reset and cross 0 an unbounded number of times because of time density.

Type-2 runs Let $\rho = (\ell_0, \nu_0), (t_1, e_1), (\ell_1, \nu_1), \ldots, (t_n, e_n), (\ell_n, \nu_n)$ be a type-1 run s.t. $\mathsf{duration}(\rho) \leq \mathbf{T}$. Let S_ρ be the set of all $0 < i \leq n$ s.t:

$$\exists x \in X : \big(\lfloor \nu_{i-1}(x) \rfloor \neq \lfloor \nu_i(x) \rfloor \big) \text{ or } \big(\lfloor \nu_{i-1}(x) \rfloor > 0 \text{ and } 0 = \langle \nu_{i-1}(x) \rangle < \langle \nu_i(x) \rangle \big)$$

where $\lfloor x \rfloor$ 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 $(0, 1)$ or from $(0, 1)$ to 0: such transitions are not recorded in S_ρ. Since each variable crosses a strictly positive integer value at most once along the *type-1 run* ρ, $|S_\rho|$ can be bounded:

Lemma 4. *For all type-1 run ρ: $|S_\rho| \leq 3 \times |X|$.*

Had we recorded in S_ρ the indices of the transitions from (ℓ, ν) to (ℓ', ν') 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_ρ by a value independent from $|\rho|$. Indeed, in any time interval, the density of time allows a variable to be reset and increase an arbitrary number of times.

Let us now explain how we split type-1 runs into type-2 runs. We first consider an example. Consider an RHA with two variables x, y (with rate 1) and one of its runs $\rho = (\ell_0, 2.1, .7) \xrightarrow{.4, e_1} (\ell_1, 2.5, 1.1) \xrightarrow{.1, e_2} (\ell_2, 2.6, 1.2) \xrightarrow{.1, e_3} (\ell_3, 0, 1.3) \xrightarrow{.1, e_4} (\ell_4, .1, 1.4) \xrightarrow{.1, e_5} (\ell_3, 0, 1.5)$, and where e_3 and e_5 reset x. Then $S_\rho = \{1, 3\}$ because y changes its integral part from $(\ell_0, 2.1, .7)$ to $(\ell_1, 2.5, 1.1)$ and x is reset by e_3 and changes its integral part. Also, $\{4, 5\} \cap S_\rho = \emptyset$ as x and y stay resp. in $[0, 1)$ and $(1, 2)$. Then, ρ is split in 5 parts: first $\rho_0 = (\ell_0, 2.1, .7)$; then $\rho'_1 = (\ell_0, 2.1, .7) \xrightarrow{.4, e_1} (\ell_1, 2.5, 1.1)$; then $\rho_1 = (\ell_1, 2.5, 1.1) \xrightarrow{.1, e_2} (\ell_2, 2.6, 1.2)$; then $\rho'_2 = (\ell_2, 2.6, 1.2) \xrightarrow{.1, e_3} (\ell_3, 0, 1.3)$ and $\rho_2 = (\ell_3, 0, 1.3) \xrightarrow{.1, e_4} (\ell_4, .1, 1.4) \xrightarrow{.1, e_5} (\ell_3, 0, 1.5)$.

Formally, assume $\rho = s_0, (t_1, e_1), s_1, \ldots, (t_n, e_n), s_n$, and $S_\rho = \{p_1, \ldots, p_k\}$, with $p_1 \leq p_2 \leq \cdots \leq p_k$. Then, we let $\rho_0, \rho_1, \ldots, \rho_k$ be the sub-runs s.t.: $\rho = \rho_0 \cdot s_{p_1 - 1}, (t_{p_1}, e_{p_1}), s_{p_1} \cdot \rho_1 \cdot s_{p_2 - 1}, (t_{p_2}, e_{p_2}), s_{p_2}, \ldots, s_{p_k - 1}, (t_{p_k}, e_{p_k}), s_{p_k} \cdot \rho_k$. Each ρ_i is called a *type-2 run*, and can be empty. In the example above, ρ_1 and ρ_2 are type-2 runs. The next lemma summarises the properties of this construction:

Lemma 5. *Let ρ be a type-1 run of* $\mathsf{R}(\mathcal{H})$ *with* $\mathsf{duration}(\rho) \leq \mathbf{T}$. *Then, ρ is split into: $\rho_0 \cdot \rho'_1 \cdot \rho_1 \cdot \rho'_2 \cdot \rho_2 \cdots \rho'_k \cdot \rho_k$ where each ρ_i is a type-2 run; $k \leq 3 \times |X|$; $|\rho'_i| = 1$ for all $1 \leq i \leq k$; and for all $1 \leq i \leq k$: $\rho_i = (\ell_0, \nu_0), (t_1, e_1), \ldots, (t_n, e_n), (\ell_n, \nu_n)$ implies that, for all $x \in X$: (i)* either *there is $a \in \mathbb{N}^{>0}$ s.t. for all $0 \leq j \leq n$: $\nu_j(x) = a$ and x is not reset along ρ_i; (ii)* or *for all $0 \leq j \leq n$: $\nu_j(x) \in (a, a+1)$ with $a \in \mathbb{N}^{>0}$ and x is not reset along ρ_i; (iii)* or *for all $0 \leq j \leq n$: $\nu_j(x) \in [0, 1)$.*

Observe that in the last case (i.e., $x \in [0, 1)$), the number of resets cannot be bounded *a priori*. For the sake of clarity, let us summarise the construction so far:

Lemma 6. *Each type-0 run of* $\mathsf{R}(\mathcal{H})$ *can be decomposed into k type-2 runs with $k \leq 3 \times (T \times \mathrm{rmax} + 1) \times |X|$.*

Contraction of type-2 runs We finish the construction by defining formally the contraction operator and establishing its properties. The formal definition follows the intuition sketched at the beginning of the section (see Fig. 2). Let $\pi = (t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$ be a timed path, let $\ell_0 = \mathsf{src}(e_1)$, and, for all $1 \leq i \leq n$: $\ell_i = \mathsf{trg}(e_i)$. Assume there are $0 \leq i < j < n$ and a function $h : \{i+1, \ldots, j-1\} \mapsto \{0, \ldots, i-1\}$ s.t. (i) $\ell_i = \ell_j$ and (ii) for all $i < p < j$: $\ell_p = \ell_{h(p)}$. Then, we let $\mathsf{Cnt}(\pi) = \ell'_0, (t'_1, e'_1), \ldots, \ell'_m$ where: (i) $m = n - (j - i)$; (ii) for all $0 \leq p \leq i$: $\ell'_p = \ell_p$; (iii) for all $1 \leq p \leq i$: $e'_p = e_p$ and $t'_p = t_p + \sum_{k \in h^{-1}(p-1)} t_{k+1}$; (iv) $e'_{i+1} = e_{j+1}$; (v) $t'_{i+1} = t_{i+1} + t_{j+1}$; and (vi) for all $i + 1 < p \leq m$: $\ell'_p = \ell_{p+j-i}$ and $(t'_p, e'_p) = (t_{p+j-i}, e_{p+j-i})$.

Then, given a timed path π, we let $\mathsf{Cnt}^0(\pi) = \pi$, $\mathsf{Cnt}^i(\pi) = \mathsf{Cnt}\left(\mathsf{Cnt}^{i-1}(\pi)\right)$ for any $i \geq 1$, and $\mathsf{Cnt}^*(\pi) = \mathsf{Cnt}^k(\pi)$ where k is the least value such that $\mathsf{Cnt}^k(\pi) = \mathsf{Cnt}^{k+1}(\pi)$. Note that $\mathsf{Cnt}^*(\pi)$ always exists since π is finite, and since, for all π: either $|\mathsf{Cnt}(\pi)| < |\pi|$ or $\mathsf{Cnt}(\pi) = \pi$. Moreover, the length of $\mathsf{Cnt}^*(\pi)$ is always bounded by a value *that does not depend on* $|\pi|$.

Lemma 7. *For all timed path π:* $|\mathsf{Cnt}^*(\pi)| \leq |\mathrm{Loc}|^2 + 1$.

Let us now lift the definition of the contraction operator to *runs* of type-2. To this end, we first need to further split type-2 runs into type-3 runs by splitting type-2 runs according to the first and last resets (if they exist) of each variable. 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 $FR_\rho = \{i \mid \exists x \in Y_i \text{ and } \forall 0 \leq j < i : x \notin Y_j\}$ and $LR_\rho = \{i \mid \exists x \in Y_i \text{ and } \forall i < j \leq n : x \notin Y_j\}$ be respectively the set of edge indices where a variable is reset for the first (last) in ρ. 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 ρ s.t. $\rho = \rho_0 \cdot s_{p(1)-1}, (t_{p(1)}, e_{p(1)}), s_{p(1)} \cdot \rho_1 \cdots s_{p(k)-1}, (t_{p(k)}, e_{p(k)}), s_{p(k)} \cdot \rho_k$. Note that each type-2 is split into at most $2 \times |X| + 1$ type-3 runs (i.e., $k \leq 2 \times |X|$). We can now define the contraction of ρ: $\mathsf{Cnt}(\rho) = \mathsf{Run}\left(\mathsf{first}(\rho), \pi_{\mathsf{Cnt}(\rho)}\right)$, where: $\pi_{\mathsf{Cnt}(\rho)} = \mathsf{Cnt}^*(\mathsf{TPath}(\rho_0)), (t_{p(1)}, e_{p(1)}), \mathsf{Cnt}^*(\mathsf{TPath}(\rho_1)), (t_{p(2)}, e_{p(2)}), \ldots, (t_{p(k)}, e_{p(k)}), \mathsf{Cnt}^*(\mathsf{TPath}(\rho_k))$. Equipped with this definition, we can show that $\mathsf{Cnt}(\rho)$ is not only of *bounded length*, but is also *equivalent* to the original type-2 run ρ, in the following sense:

Proposition 8. *For all type-2 runs* ρ, $\mathsf{Cnt}(\rho) \neq \bot$, $\mathsf{first}(\mathsf{Cnt}(\rho)) = \mathsf{first}(\rho)$, $\mathsf{last}(\mathsf{Cnt}(\rho)) = \mathsf{last}(\rho)$ *and* $|\mathsf{Cnt}(\rho)| \leq 8 \times |\mathrm{Loc}|^2 \times |X|$.

Proof (sketch). We sketch the proof assuming $\mathcal{H}$ has only one variable x with non-negative rate (the arguments generalise easily). Let $\rho = (\ell_0, \nu_0) \xrightarrow{t_1, e_1} (\ell_1, \nu_1) \cdots \xrightarrow{t_n, e_n} (\ell_n, \nu_n)$ be a type-2 run, $\pi = \mathsf{TPath}(\rho)$, $\pi' = \mathsf{Cnt}^*(\pi)$ and $\rho' = \mathsf{Cnt}(\rho)$. Observe that $\mathsf{duration}(\pi') = \mathsf{duration}(\rho)$, and that $\mathsf{Effect}(\pi') = \mathsf{Effect}(\pi)$. Assume *first* that x is never reset along ρ, and that $\nu_0(x) \notin [0, 1)$. Then, all valuations of x along ρ are in the same interval $[a, a]$ or $(a, a+1)$ for $a \geq 1$, by Lemma 5 (thus the issue of case 1 above is ruled out). In this case, all the guards are still satisfied in π', and $\rho' \neq \bot$. Finally, assuming $\mathsf{last}(\rho') = (\ell_n, \nu'_n)$, we have $\nu'_n(x) = \nu_0(x) + \mathsf{Effect}(\pi')(x) = \nu_0(x) + \mathsf{Effect}(\pi)(x) = \nu_n(x)$ because x is not reset along ρ. *Second*, assume x is never reset along ρ and that $\nu_0(x) \in [0, 1)$. In this case, we have to rule out an additional difficulty. Let k, j be s.t. $k < j$,. $\nu_j(x) = 0$, $t_{j+1} = 0$, e_{j+1} has guard '$x = 0$' and $t_k > 0$: we must show that $\ell_k \neq \ell_j$ (otherwise the delay $t_k > 0$ could be 'reported' on ℓ_j, and the guard of e_j would not be satisfied, this is the problem identified in case 4). $\ell_k \neq \ell_j$ holds because, by construction of $\mathsf{Reg}(\mathcal{H})$, $\ell_k = (\ell, \mathbf{0}^=)$ and $\ell_j = (\ell', \mathbf{0}^+)$ for some ℓ, ℓ', because x is null when leaving ℓ_k, but not when leaving ℓ_j. *Third*, assume x is reset along ρ, hence x takes values in $[0, 1)$ only along ρ, by Lemma 5. Let j, k be s.t. $j < k$ and e_j (resp. e_k) is the first (last) edge to reset x along ρ. Then, by definition, $\pi' = \mathsf{Cnt}^*(\pi[0 : j-1]), (t_j, e_j), \mathsf{Cnt}^*(\pi[j+1, k-1])(t_k, e_k)\mathsf{Cnt}^*(\pi[k+1, n])$. In $\mathsf{Cnt}^*(\pi[0 : j-1])$ and $\mathsf{Cnt}^*(\pi[k+1, n])$, x is not reset and takes values in $[0, 1)$, thus $\mathsf{Cnt}^*(\pi[0 : j-1])$ and $\mathsf{Cnt}^*(\pi[k+1, n])$ yield genuine runs, by the same arguments as above. If x is reset in $\pi[j+1, k-1]$, this is not the first reset along π, so we avoid the issue of case 2. Thanks to the $\mathbf{0}^+$ and $\mathbf{0}^=$ regions, we are sure that the '$x = 0$' guards are still satisfied in $\mathsf{Cnt}^*(\pi[j+1, k-1])$, so it yields a genuine run. Thus, $\rho' \neq \bot$. Note however that the value of x after firing $\mathsf{Cnt}^*(\pi[0 : j-1]), (t_j, e_j), \mathsf{Cnt}^*(\pi[j+1, k-1])$ might not be the same as when firing $\pi[0 : k-1]$. Yet, this does not prevent from firing e_k. Moreover, the value of x at the end of the run is preserved (i.e., we avoid the issue of case 3 above): if $\mathsf{last}(\rho') = (\ell_n, \nu'_n)$, then $\nu'_n(x) = \mathsf{Effect}(\mathsf{Cnt}^*(\pi[k+1, n]))(x)$ (again because x is not reset along $\pi[k+1, n]$) with $\mathsf{Effect}(\mathsf{Cnt}^*(\pi[k+1, n]))(x) = \mathsf{Effect}(\pi[k+1, n])(x) = \nu_n(x)$. □

We obtain Theorem 1 thanks to Proposition 8, Lemma 6 and the definition of $\mathsf{Reg}(\mathcal{H})$.

Rectangular rates Let us now briefly explain how we can adapt the previous construction to cope with non-singular rates. Let us first notice that for all MHA $\mathcal{H}$, $\mathsf{R}(\mathcal{H})$ is still well-defined. Then, we adapt the definition of timed path as follows. A timed path is a sequence $(t_1, R_1, e_1) \cdots (t_n, R_n, e_n)$, where each $R_i : X \mapsto \mathbb{R}$ 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 and Effect to take those rates into account and retain the properties needed to prove Theorem 2. 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, $R = \frac{\sum_{i=1}^{n} t_i \times R_i}{t}$. Note that we rely on the convexity of the invariants and rates in an RHA to ensure that this construction is correct.

4 Time-bounded reachability is NEXPTIME-complete

In this section, we establish our main result:

Theorem 9. *Time-bounded reachability for MHA is* NEXPTIME *complete.*

An NEXPTIME *algorithm* Recall that an instance of the time-bounded reachability problem is of the form $(\mathcal{H}, \ell, \mathbf{T})$, where $\mathcal{H}$ is an MHA, ℓ is a location, and $\mathbf{T}$ is a time bound (expressed in binary). We establish membership in NEXPTIME by giving a non-deterministic algorithm that runs in time exponential in the size of $(\mathcal{H}, \ell, \mathbf{T})$ in the worst case. The algorithm *guesses* a sequence of edges $\mathcal{E} = e_0 e_1 \dots e_n$ of $\mathcal{H}$ such that $n+1 \leq F(\mathcal{H}, \mathbf{T})$ and $\mathsf{trg}\,(e_n) = \ell$ and builds a linear constraint $\Phi(\mathcal{E})$, that expresses all the properties that must be satisfied by a run following the sequence of edges $\mathcal{E}$ (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 $\Phi(\mathcal{E})$ is satisfiable and returns 'yes' iff it is the case.

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

NEXPTIME-*hardness* To establish the NEXPTIME-hardness, we encode the membership problem of non-deterministic exponential time Turing machines (NExpTM for short) to time-bounded reachability for SMHA. An NExpTM is a tuple $M = (Q, \Sigma, \Gamma, q_0, \delta, F, \xi)$ where Q is the (nonempty and finite) set of control states, $\Sigma = \{0, 1\}$ is the (finite) input alphabet[2], $\Gamma = \{\sharp, 0, 1\}$ is the (finite) alphabet of the tape, where $\sharp$ 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 $\xi = \mathcal{O}\left(2^{p(n)}\right)$ (for some polynomial p), is an exponential function to bound the execution length.

A state of M is a triple (q, w_1, w_2), where $q \in Q$, and $w_1, w_2 \in \Gamma^*$ are resp. the content to the left (to the right and below) of the reading head, excluding the trailing sequence of $\sharp$. We rely on the standard semantics for NExpTM: for example, (q_1, a, b, L, q_2) means 'when in q_1 and a is below the head, replace a by b, move the head to the left (L) and go to q_2'. We write $(q, w_1, w_2) \triangleright (q', w'_1, w'_2)$ when there is a transition from (q, w_1, w_2) to (q', w'_1, w'_2). An execution of M on input w is a finite sequence of states $c_0 c_1 \dots c_n$ such that: (i) $n \leq \xi(|w|)$; (ii) $c_0 = (q_0, \varepsilon, w \cdot \sharp^{\xi(|w|)-|w|})$; and (iii) for all $0 \leq i < n$: $c_i \triangleright c_{i+1}$. It is *accepting* iff $c_n = (q, w_1, w_2)$ with $q \in F$.

Let us show how to encode all executions of M into the executions of an SMHA $\mathcal{H}_M$. We encode the words w_1 and w_2 as pairs of rational values (l_1, c_1) and (l_2, c_2) where $l_i = \frac{1}{2^{|w_i|}}$ encodes the length of the word w_i by a rational number in $[0, 1]$, and c_i encodes w_i as follows. Assume $w_1 = \sigma_0 \sigma_1 \dots \sigma_n$. Then, we let $c_1 = \mathsf{Val}^{\leftarrow}(w_1) =$

[2] Having $\Sigma = \{0, 1\}$ and $\Gamma = \Sigma \cup \{\sharp\}$ is without loss of generality.

$\sigma_n \cdot \frac{1}{2} + \sigma_{n-1} \cdot \frac{1}{4} + \cdots + \sigma_0 \cdot \frac{1}{2^{n+1}}$. Intuitively, c_1 is the value which is represented in binary by $0.\sigma_n\sigma_{n-1}\cdots\sigma_0$, i.e., w_1 is the binary encoding of the fractional part of c_1 with the most significant bit in the rightmost position. For instance, if $w_1 = 001010$ then $\mathsf{Val}^{\leftarrow}(w_1) = 0 \cdot \frac{1}{2} + 1 \cdot \frac{1}{4} + 0 \cdot \frac{1}{8} + 1 \cdot \frac{1}{16} + 0 \cdot \frac{1}{32} + 0 \cdot \frac{1}{64} = 0.3125$, and so w_1 is encoded as the pair $(\frac{1}{64}, 0.3125)$. Note that we need to remember the actual length of the word w_1 because the function $\mathsf{Val}^{\leftarrow}(\cdot)$ ignores the leading 0's (for instance, $\mathsf{Val}^{\leftarrow}(001010) = \mathsf{Val}^{\leftarrow}(1010)$). Symmetrically, if $w_2 = \sigma_0\sigma_1 \ldots \sigma_n$, we let $c_2 = \mathsf{Val}^{\rightarrow}(w_2) = \sigma_0 \cdot \frac{1}{2} + \sigma_1 \cdot \frac{1}{4} + \cdots + \sigma_n \cdot \frac{1}{2^{n+1}}$ (i.e., σ_0 is now the most significant bit). Then a state (q, w_1, w_2) of the NExpTM 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 for 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 NExpTM. Assume $w_1 = w_0^1 w_2^1 \cdots w_n^1$ and $w_2 = w_0^2 w_2^2 \cdots w_k^2$. To *read the letter under the head* we need to test the value of the bit w_0^2. Clearly, $w_0^2 = 1$ iff $l_2 \leq 1/2$, and $c_2 \geq \frac{1}{2}$; $w_0^2 = 0$ iff $l_2 \leq 1/2$, and $c_2 < \frac{1}{2}$ and $w_0^2 = \sharp$ iff $l_2 = 1$ (which corresponds to $w_2 = \varepsilon$). To *test whether the head is in the leftmost cell of the tape* we must check whether $w_1 = \varepsilon$, i.e. $l_1 = 1$. To *read the letter at the left of the head* (assuming that $w_1 \neq \varepsilon$) we must test the value of the bit w_n^1. Clearly, $w_n^1 = 1$ iff $c_1 \geq \frac{1}{2}$ and $w_n^1 = 0$ 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's 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 NExpTM transition. *To append a* 1 *to the right* of w_1, we let $l_1' = \frac{1}{2} \times l_1$. We let $c_1' = \frac{1}{2}$ if $l_1 = 1$ (i.e. w_1 was empty) and $c_1' = \frac{1}{2} \times c_1 + \frac{1}{2}$ otherwise. *To append a* 0 *to the right of* w_1, we let $l_1' = \frac{1}{2} \times l_1$ and $c_1' = \frac{1}{2} \times c_1$. *To delete a* 0 *from the rightmost position of* w_1, we let $l_1' = 2 \times l_1$, $c_1' = 2 \times c_1$. *To delete a* 1 *from the rightmost position of* w_1, we let $l_1' = 2 \times l_1$, and $c_1' = (c_1 - \frac{1}{2}) \times 2$. In addition, note that we can flip the leftmost bit of w_2 by adding or subtracting $1/2$ from c_2 (this is necessary when updating the value under the head). Thus, the operations that we need to be able to perform on c_1, l_1, c_2 and l_2 are: to multiply by 2, divide by 2, increase by $\frac{1}{2}$ and decrease by $\frac{1}{2}$, while keeping untouched the value of all the other variables. Fig. 3 exhibits four gadgets to perform these operations. Note that these gadgets can be constructed in polynomial time, execute in 1 time unit time and bear only singular rates.

We claim that all transitions of M can be simulated by combining the gadgets in Fig. 3 and the tests described above. For instance, consider the transition: $(q_1, 1, 0, L, q_2)$. It is simulated as follows. First, we check that the reading head is not at the leftmost position of the tape by checking that $l_1 < 1$. Second, we check that the value below the reading head is equal to 1 by testing that $l_2 < 1$ and $c_2 \geq \frac{1}{2}$. Third, we change the value below the reading head from 1 to 0 by subtracting $\frac{1}{2}$ from c_2 using an instance of gadget (iii) in Fig. 3. And finally, we move the head one cell to the left. This is performed by testing the bit on the left of the head, deleting it from w_1 and appending it to the left of w_2, by the operations described above. All other transitions can be simulated similarly. Note that, to simulate one NExpTM transition, we need to perform several tests (that

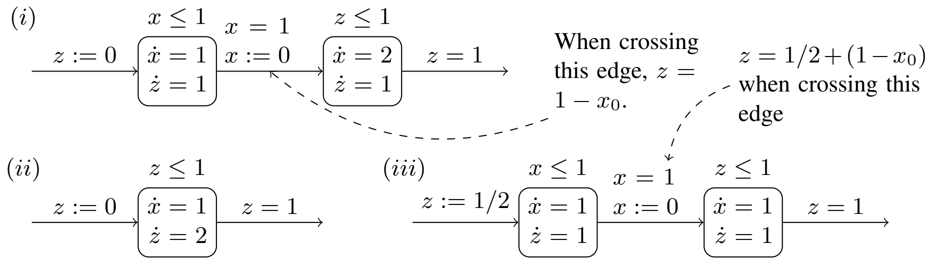

Fig. 3. Gadgets (i) for multiplication by 2, (ii) adding $\frac{1}{2}$ and (iii) subtracting $\frac{1}{2}$. The rates of the $y \notin \{x, z\}$ is 0. Gadget (i) can be modified to divide by 2, by swapping the rates of x and z in the second location. x_0 is the value of x when entering the gadget.

carry out in 0 time units) and to: (i) update the bit under the reading head, which takes 1 time unit with our gadgets; (ii) remove one bit from the right of w_1 (resp. left of w_2), which takes at most 3 time units and (iii) append this bit to the left of w_2 (right of w_1), which takes at most 3 time units. We conclude that each NExpTM transition can be simulate in at most 7 time units. Thus M has an accepting execution on word w (of length at most $\xi(|w|)$ iff $\mathcal{H}_M$ has an execution of duration at most $\mathbf{T} = 7 \cdot \xi(|w|)$ that reaches a location encoding an accepting control state of M. This sets the reduction.

5 Computing all the states reachable within $\mathbf{T}$ time units

Let us now show that Theorem 2 (lifted to MHA) implies that, in an MHA, we can compute a *symbolic representation* of the set of states reachable within $\mathbf{T}$ time units. We show, by means of two examples, that this information can be used to verify meaningful properties of MHA, in particular when time-*unbounded* fixed points do not terminate.

Post *and* Pre Let s be a state of an MHA with set of edges Edges. We let $\mathsf{Post}(s) = \{s' \mid \exists e \in \text{Edges}, t \in \mathbb{R}^+ : s \xrightarrow{t,e} s'\}$ and $\mathsf{Pre}(s) = \{s' \mid \exists e \in \text{Edges}, t \in \mathbb{R}^+ : s' \xrightarrow{t,e} s\}$. We further let $\mathsf{Reach}^{\le \mathbf{T}}(s) = \{s' \mid \exists \pi : s \xrightarrow{\pi} s' \wedge \mathsf{duration}(\pi) \le \mathbf{T}\}$, and $\mathsf{coReach}^{\le \mathbf{T}}(s) = \{s' \mid \exists \pi : s' \xrightarrow{\pi} s \wedge \mathsf{duration}(\pi) \le \mathbf{T}\}$ be respectively the set of states that are reachable from s (that can reach s) within $\mathbf{T}$ time units. We extend all those operators to sets of states in the obvious way. Our aim in this section is to compute effective representations of $\mathsf{Reach}^{\le \mathbf{T}}(s)$ and $\mathsf{coReach}^{\le \mathbf{T}}(s)$, using fixed points.

Symbolic states To manipulate potentially infinite sets of MHA states, we need a symbolic representation that is manipulable algorithmically. We rely on the notion of *symbolic states* introduced as an *algebra of regions* in [11]. To manipulate sets of valuations, we use formulas of $(\mathbb{R}, 0, 1, +, \le)$, i.e. the first-order logic of the reals[3], with the constants 0 and 1, the usual order $\le$ and addition $+$ [11]. Recall that the satisfiability problem for that logic is decidable [4] and that it admits effective quantifier

[3] In practice, those formulas can be manipulated as finite unions of convex polyhedra for which there exist efficient implementations, see [3] for example.

elimination. Furthermore, all guards of an MHA can be represented by a formula from $(\mathbb{R}, 0, 1, +, \leq)$ ranging over X. Let Ψ be a formula of $(\mathbb{R}, 0, 1, +, \leq)$, and let ν be a valuation of the free variables of Ψ. Then we write $\nu \models \Psi$ iff ν satisfies Ψ, and we let $[\![\Psi]\!]$ be the set off all valuations ν such that $\nu \models \Psi$. To emphasise the fact that a formula Ψ ranges over the set of variables X, we sometimes denote it by $\Psi(X)$.

Then a *symbolic state* of an MHA $\mathcal{H}$ with set of variables X is a function R mapping each location ℓ of $\mathcal{H}$ to a quantifier free formula of $(\mathbb{R}, 0, 1, +, \leq)$ with free variables in X, representing sets of valuations for the variables in ℓ. Formally, R represents the set of MHA states $[\![R]\!] = \{(\ell, \nu) \mid \nu \in [\![R(\ell)]\!]\}$. By abuse of notation, we assume that any formula Φ of $(\mathbb{R}, 0, 1, +, \leq)$ denotes the function f such that $f(\ell) = \Phi$ for all ℓ. Clearly, given symbolic states R_1 and R_2, one can compute symbolic states $R_1 \vee R_2$ and $R_1 \wedge R_2$ representing resp. $[\![R_1]\!] \cup [\![R_2]\!]$ and $[\![R_1]\!] \cap [\![R_2]\!]$; and one can test whether $[\![R_1]\!] = [\![R_2]\!]$ [11]. It is also possible (see details in the appendix) to compute Post and Pre symbolically: we let $\mathsf{post}^\sharp$ and $\mathsf{pre}^\sharp$ be effective operators, returning symbolic states, s.t. for all symbolic states R: $[\![\mathsf{post}^\sharp(R)]\!] = \mathsf{Post}([\![R]\!])$ and $[\![\mathsf{pre}^\sharp(R)]\!] = \mathsf{Pre}([\![R]\!])$.

Time-bounded forward and backward fixpoints Let $\mathcal{H}$ be an MHA with set of variables X, and let $\mathbf{T} \in \mathbb{N}$ 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 S be a *set of states* of $\mathcal{H}$. Then the sets $\mathsf{Reach}^{\leq \mathbf{T}}(S)$ and $\mathsf{coReach}^{\leq \mathbf{T}}(S)$ can be defined by means of fixed point equations: $\mathsf{Reach}^{\leq \mathbf{T}}(S) = \mu Y \cdot ((S \cup \mathsf{Post}(Y)) \cap [\![0 \leq t \leq \mathbf{T}]\!])$ and $\mathsf{coReach}^{\leq \mathbf{T}}(S) = \mu Y \cdot ((S \cup \mathsf{Pre}(Y)) \cap [\![0 \leq t \leq \mathbf{T}]\!])$. This observation forms the basis of our algorithm for computing symbolic states representing $\mathsf{Reach}^{\leq \mathbf{T}}([\![R]\!])$ and $\mathsf{coReach}^{\leq \mathbf{T}}([\![R]\!])$ for some symbolic state $R(X)$. Let $(F_i)_{i \geq 0}$ and $(B_i)_{i \geq 0}$ be the sequences of symbolic states defined as follows: $F_0 = B_0 = R(X) \wedge (0 \leq t \leq \mathbf{T})$; and for all $i \geq 1$: $F_i = \mathsf{post}^\sharp(F_{i-1}) \wedge (0 \leq t \leq \mathbf{T}) \vee F_{i-1}$ and $B_i = \mathsf{pre}^\sharp(B_{i-1}) \wedge (0 \leq t \leq \mathbf{T}) \vee B_{i-1}$. Note that, for all $i \geq 1$, F_i (resp. B_i) can be computed from F_{i-1} (B_{i-1}).

Proposition 10. *For all MHA $\mathcal{H}$, all symbolic states R and all time bound $\mathbf{T}$, there are k and ℓ such that $0 \leq k, \ell \leq F(\mathcal{H}, \mathbf{T})$, $[\![F_k]\!] = [\![F_{k+1}]\!] = \mathsf{Reach}^{\leq \mathbf{T}}([\![R]\!])$ and $[\![B_\ell]\!] = [\![B_{\ell+1}]\!] = \mathsf{coReach}^{\leq \mathbf{T}}([\![R]\!])$. Computing F_k and B_ℓ takes at most doubly exponential time.*

By Theorem 9, this deterministic algorithm can be considered optimal (unless NEXPTIME=EXPTIME). Let us show, by two examples, the usefulness of our approach.

Example 1: Leaking gas burner With this example, the *time-unbounded* forward fixed-point computation does not terminate, in contrast to the time-bounded fixed-point computation. The gas burner in the example 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. An MHA modeling this gas burner [1] is given in Fig. 4. Stopwatch y and clock t are used resp. to measure the leakage time and the total elapsed time. One can show using *backward* analysis that, in any time

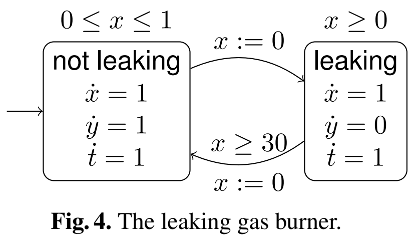

Fig. 4. The leaking gas burner.

interval of at least 60 seconds, the leakage time is at most 5% of the elapsed time [8]. The backward fixpoint is obtained after 7 iterations but the forward does not terminate.

Using forward time-bounded reachability analysis we can prove that, in all time intervals of fixed length $T \geq 60$, the leakage time is at most $\frac{T}{20}$. To prove that this property holds in *all* time intervals, we first compute, using the algorithm described above (see Proposition 10), $\mathsf{Reach}^{\leq 60}([\![R]\!])$, where $[\![R]\!] = \{(\ell, v) \mid \ell = \mathsf{leaking} \text{ implies } 0 \leq v(x) \leq 1\}$, i.e. R represents all possible states of the system. HYTECH computes $\mathsf{Reach}^{\leq 60}([\![R]\!])$ after 5 iterations of the forward time-bounded fixpoint. Then, we check that '$t = 60$ implies $y \leq \frac{60}{20} = 3$' holds, in all states of $\mathsf{Reach}^{\leq 60}([\![R]\!])$.

Example 2: bounded invariant Let us come back to the RHA$^{\geq 0}$ of Fig. 1 (left). Notice that all variables have a bounded invariant $[0, 1]$. The forward reachability analysis of HyTech does not terminate here because the set of reachable states is not a finite union of polyhedra, see Fig. 1 (right). Yet, the time-bounded forward fixpoint terminates for all $\mathbf{T}$ by Proposition 10. This example shows that bounding the variables is not sufficient to obtain termination while performing time-bounded analysis is.

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. Springer, 1993.
2. R. Alur and D. Dill. A theory of timed automata. *TTCS*, 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), 2008.
4. S. Basu. New results on quantifier elimination over real closed fields and applications to constraint databases. *J. ACM*, 46(4), 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'11*, LNCS 6756. Springer, 2011.
6. T. Brihaye, L. Doyen, G. Geeraerts, J. Ouaknine, J.-F. Raskin, and J. Worrell. Time-bounded reachability for hybrid automata: Complexity and fixpoints. Technical report CoRR abs/1211.1276, Cornell University Library, arXiv.org, 2012. http://arxiv.org/abs/1211.1276.
7. T. A. Henzinger. The theory of hybrid automata. In *LICS'96*. IEEE Computer Society, 1996.
8. T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. A user guide to HYTECH. In *TACAS'95*, LNCS 1019. Springer-Verlag, 1995.
9. T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. Hytech: A model checker for hybrid systems. In *CAV'97*, LNCS 1254. Springer, 1997.
10. T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata. *JCSS*, 57(1):94–124, 1998.
11. T. A. Henzinger, R. Majumdar, and J.-F. Raskin. A classification of symbolic transition systems. *ACM Trans. Comput. Log.*, 6(1):1–32, 2005.
12. M. Jenkins, J. Ouaknine, A. Rabinovich, and J. Worrell. Alternating timed automata over bounded time. In *LICS'10*. IEEE Computer Society, 2010.
13. S. K. Jha, B. H. Krogh, J. E. Weimer, and E. M. Clarke. Reachability for linear hybrid automata using iterative relaxation abstraction. In *HSCC'07*, LNCS 4416. Springer, 2007.
14. J. Ouaknine, A. Rabinovich, and J. Worrell. Time-bounded verification. In *CONCUR'09*, LNCS 5710. Springer, 2009.