On Reachability for Hybrid Automata over Bounded Time*

Thomas Brihaye† Laurent Doyen‡ Gilles Geeraerts§
Joël Ouaknine¶ Jean-François Raskin‡ James Worrell§

April 28, 2011

Abstract

This paper investigates the time-bounded version of the reachability problem for hybrid automata. This problem asks whether a given hybrid automaton can reach a given target location within $\mathbf{T}$ time units, where $\mathbf{T}$ is a constant rational value. We show that, in contrast to the classical (unbounded) reachability problem, the timed-bounded version is *decidable* for rectangular hybrid automata provided only non-negative rates are allowed. This class of systems is of practical interest and subsumes, among others, the class of stopwatch automata. We also show that the problem becomes undecidable if either diagonal constraints or both negative and positive rates are allowed.

1 Introduction

The formalism of hybrid automata [1] is a well-established model for hybrid systems whereby a digital controller is embedded within a physical environment. The state of a hybrid system changes both through discrete transitions of the controller, and continuous evolutions of the environment. The discrete state of the system is encoded by the *location* ℓ of the automaton, and the continuous state is encoded by *real-valued variables* X evolving according to dynamical laws constraining the first derivative $\dot{X}$ of the variables. Hybrid automata have proved useful in many applications, and their analysis is supported by several tools [6, 5].

*Work supported by the projects: (i) QUASIMODO (FP7- ICT-STREP-214755), Quasimodo: "Quantitative System Properties in Model-Driven-Design of Embedded", `http://www.quasimodo.aau.dk/`, (ii) GASICS (ESF-EUROCORES LogiCCC), Gasics: "Games for Analysis and Synthesis of Interactive Computational Systems", `http://www.ulb.ac.be/di/gasics/`, (iii) Moves: "Fundamental Issues in Modelling, Verification and Evolution of Software", `http://moves.ulb.ac.be`, a PAI program funded by the Federal Belgian Government, (iv) the ARC project AUWB-2010–10/15-UMONS-3, (v) the FRFC project 2.4515.11 and (vi) a grant from the National Bank of Belgium.

†Université de Mons, Belgium
‡LSV, ENS Cachan & CNRS, France
§Université Libre de Bruxelles, Belgium
¶Oxford University Computing Laboratory, UK

A central problem in hybrid-system verification is the *reachability problem* which is to decide if there exists an execution from a given initial location ℓ to a given goal location ℓ'. While the reachability problem is undecidable for simple classes of hybrid automata (such as linear hybrid automata [1]), the decidability frontier of this problem is sharply understood [7, 8]. For example, the reachability problem is decidable for the class of initialized rectangular automata where (i) the flow constraints, guards, invariants and discrete updates are defined by rectangular constraints of the form $a \leq \dot{x} \leq b$ or $c \leq x \leq d$ (where a, b, c, d are rational constants), and (ii) whenever the flow constraint of a variable x changes between two locations ℓ and ℓ', then x is reset along the transition from ℓ to ℓ'. Of particular interest is the class of timed automata which is a special class of initialized rectangular automata [2].

In recent years, it has been observed that new decidability results can be obtained in the setting of time-bounded verification of real-time systems [10, 11]. Given a time bound $\mathbf{T} \in \mathbb{N}$, the time-bounded verification problems consider only traces with duration at most $\mathbf{T}$. Note that due to the density of time, the number of discrete transitions may still be unbounded. Several verification problems for timed automata and real-time temporal logics turn out to be decidable in the time-bounded framework (such as the language-inclusion problem for timed automata [10]), or to be of lower complexity (such as the model-checking problem for MTL [11]). The theory of time-bounded verification is therefore expected to be more robust and better-behaved in the case of hybrid automata as well.

Following this line of research, we revisit the reachability problem for hybrid automata with time-bounded traces. The *time-bounded reachability problem* for hybrid automata is to decide, given a time bound $\mathbf{T} \in \mathbb{N}$, if there exists an execution of duration less than $\mathbf{T}$ from a given initial location ℓ to a given goal location ℓ'. We study the frontier between decidability and undecidability for this problem and show how bounding time alters matters with respect to the classical reachability problem. In this paper, we establish the following results. First, we show that the time-bounded reachability problem is *decidable* for non-initialized rectangular automata when only positive rates are allowed[1]. The proof of this fact is technical and, contrary to most decidability results in the field, does not rely on showing the existence of an underlying finite (bi)simulation quotient. We study the properties of time-bounded runs and show that if a location is reachable within $\mathbf{T}$ time units, then it is reachable by a timed run in which the number of discrete transitions can be bounded. This in turn allows us to reduce the time-bounded reachability problem to the satisfiability of a formula in the first-order theory of real addition, decidable in EXPSPACE [4].

Second, we show that the time-bounded reachability problem is *undecidable* for non-initialized rectangular hybrid automata if both positive and negative rates are allowed. Third, we show that the time-bounded reachability problem is *undecidable* for initialized rectangular hybrid automata with positive singular flows if diagonal constraints in guards are allowed. These two undecidability results allow to precisely characterize the boundary between decidability and undecidability.

The undecidability results are obtained by reductions from the halting problem for

[1]This class is interesting from a practical point of view as it includes, among others, the class of stopwatch automata [3], for which unbounded reachability is undecidable.

two-counter machines. We present novel encodings of the execution of two-counter machines that fit into time-bounded executions of hybrid automata with either negative rates, or diagonal constraints.

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 $X' = \{x' \mid x \in X\}$ and $\dot{X} = \{\dot{x} \mid x \in X\}$ be the set of primed and dotted variables, corresponding respectively to variable updates and 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 $\dot{X}$ they are called *rate constraints*, and over $X \cup X'$ they are called *update constraints*. A guard or rate constraint is *rectangular* if all its constraints are rectangular. An update constraint is *rectangular* if all its constraints are either rectangular or of the form $x = x'$. We denote by $\mathcal{G}(X)$, $\mathcal{R}(X)$, $\mathcal{U}(X)$ respectively the sets of guards, rate constraints, and update constraints over X.

Linear 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*; $\mathrm{Edges} \subseteq \mathrm{Loc} \times \mathcal{G}(X) \times \mathcal{U}(X) \times \mathrm{Loc}$ 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 $\mathrm{Init} \in \mathrm{Loc}$ is an *initial location*. For an edge $e = (\ell, g, r, \ell')$, we denote by $\mathsf{src}(e)$ and $\mathsf{trg}(e)$ the location ℓ and ℓ' respectively, g is called the *guard* of e and r is the *update* (or *reset*) of e. In the sequel, we denote by rmax the maximal constant occurring in the constraints of $\{\mathrm{Rates}(\ell) \mid \ell \in \mathrm{Loc}\}$

A LHA $\mathcal{H}$ is *singular* if for all locations ℓ and for all variables x of $\mathcal{H}$, the only constraint over $\dot{x}$ in $\mathrm{Rates}(\ell)$ is of the form $\dot{x} \in I$ where I is a singular interval; it is *fixed rate* if for all variables x of $\mathcal{H}$ there exists $I_x \in \mathcal{I}$ such that for all locations ℓ of $\mathcal{H}$, the only constraint on $\dot{x}$ in $\mathrm{Rates}(\ell)$ is the constraint $\dot{x} \in I_x$. It is *multirate* if it is not fixed rate. It is *non-negative rate* if for all variables x, for all locations ℓ, the constraint $\mathrm{Rates}(\ell)$ implies that $\dot{x}$ must be non-negative.

Rectangular hybrid automata. A *rectangular hybrid automaton* (RHA) is a linear hybrid automaton in which all guards, rates, and invariants are rectangular. In this case, we view each reset r as a function $X' \mapsto \mathcal{I} \cup \{\bot\}$ that associates to each variable $x \in X$ either an interval of possible reset values $r(x)$, or $\bot$ when the value of the variable x remains unchanged along the transition. When it is the case that $r(x)$ is either $\bot$ or a singular interval for each x, we say that r is *deterministic*. In the case of RHA, we can also view rate constraints as functions $\mathrm{Rates} : \mathrm{Loc} \times X \to \mathcal{I}$ that associate to each location ℓ and each variable x an interval of possible rates $\mathrm{Rates}(\ell)(x)$. A rectangular hybrid automaton $\mathcal{H}$ is *initialized* if for every edge (ℓ, g, r, ℓ') of $\mathcal{H}$, for every $x \in X$, if

$\mathrm{Rates}(\ell)(x) \neq \mathrm{Rates}(\ell')(x)$ then $r(x) \neq \bot$, i.e., every variable whose rate constraint is changed must be reset.

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

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. 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 (ℓ', ν') if $e = (\ell, g, r, \ell') \in \mathrm{Edges}$, $\nu \models g$, $\nu'(x) = \nu(x)$ for all x s.t. $r(x) = \bot$, and $\nu'(x) \in r(x)$ for all x s.t. $r(x) \neq \bot$; 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 (ℓ, ν') if there exists 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 *cycle* is a path $e_1, e_2, \ldots, e_n$ such that $\mathsf{trg}(e_n) = \mathsf{src}(e_1)$. A cycle $e_1, e_2, \ldots, e_n$ is *simple* if $\mathsf{src}(e_i) \neq \mathsf{src}(e_j)$ for all $i \neq j$. 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 $0 \leq i \leq n$. We lift the notions of cycle and simple cycle to the timed case accordingly. Given a timed path $\pi = (t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$, we denote by $\pi[i:j]$ (with $1 \leq i \leq j \leq n$) the timed path $(t_i, e_i), \ldots, (t_j, e_j)$.

A *run* in $\mathcal{H}$ is a sequence $s_0, (t_0, e_0), s_1, (t_1, e_1), \ldots, (t_{n-1}, e_{n-1}), s_n$ such that:

- $(t_0, e_0), (t_1, e_1), \ldots, (t_{n-1}, e_{n-1})$ is a timed path in $\mathcal{H}$, and
- for all $1 \leq i < n$, there exists a state s'_i of $\mathcal{H}$ with $s_i \xrightarrow{t_i} s'_i \xrightarrow{e_i} s_{i+1}$.

Given a run $\rho = s_0, (t_0, e_0), \ldots, s_n$, let $\mathsf{first}(\rho) = s_0 = (\ell_0, \nu_0)$, $\mathsf{last}(\rho) = s_n$, $\mathsf{duration}(\rho) = \sum_{i=1}^{n-1} t_i$, and $|\rho| = n + 1$. We say that ρ is (i) *strict* if $t_i > 0$ for all $1 \leq i \leq n-1$; (ii) *k-variable-bounded* (for $k \in \mathbb{N}$) if $\nu_0(x) \leq k$ for all $x \in X$, and $s_i \xrightarrow{t_i} (\ell_i, \nu_i)$ implies that $\nu_i(x) \leq k$ for all $0 \leq i \leq n$; (iii) **T**-*time-bounded* (for $\mathbf{T} \in \mathbb{N}$) if $\mathsf{duration}(\rho) \leq \mathbf{T}$.

Note that a unique timed path $\mathsf{TPath}(\rho) = (t_0, e_0), (t_1, e_1), \ldots, (t_{n-1}, e_{n-1})$, is associated to each run $\rho = s_0, (t_0, e_0), s_1, \ldots, (t_{n-1}, e_{n-1}), s_n$. Hence, we sometimes abuse 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, if such a run exists it is necessarily unique *when the automaton is singular and all resets are deterministic*. In that case, 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 $\bot$ otherwise.

Time-bounded reachability problem for LHA. While the reachability problem asks to decide the existence of any timed run that reaches a given goal location, we are only interested in runs having bounded duration.

Problem 1 (Time-bounded reachability problem) *Given an LHA* $\mathcal{H} = (X, \mathrm{Loc}, \mathrm{Edges}, \mathrm{Rates}, \mathrm{Inv}, \mathrm{Init})$*, 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* $\rho = (\mathrm{Init}, \vec{0}) \xrightarrow{\pi} (\mathrm{Goal}, \cdot)$ *of* $\mathcal{H}$ *with* $\mathsf{duration}(\rho) \leq \mathbf{T}$.

In the following table, we summarize the known facts regarding decidability of the reachability problem for LHA, along with the results on time-bounded reachability that we prove in the rest of this paper. Note that decidability for initialized rectangular hybrid automata (IHRA) follows directly from [7]. We show decidability for (non-initialized) RHA that only have non-negative rates in Section 3. The undecidability of the time-bounded reachability problem for RHA and LHA is not a consequence of the known results from the literature and require new proofs that are given in Section 4.

HA classes	Reachability	Time-Bounded Reachability
LHA	U [1]	**U** (see Section 4)
RHA	U [7]	**U** (see Section 4)
non-negative rates RHA	U [7]	**D** (see Section 3)
IRHA	D [7]	D [7]

Example of time bounded reachability Let $\mathcal{H}$ be the hybrid automaton of Fig. 1 with the convention that the transition starting from ℓ_i and ending in ℓ_j is denoted e_{ij}. Although not explicitly stated on the figure, we assume that all the locations are equipped with the invariant $(x \leq 1) \wedge (y \leq 1)$. As this automaton uses only rectangular constraints and positive rates, it is in the class for which we show the decidability of the time-bounded reachability problem (see Section 3). Note that it is non-initialized as, for example, variable y is not reset from location ℓ_0 to location ℓ_1 while its rate is changing, and it is singular, diagonal-free, and multirate.

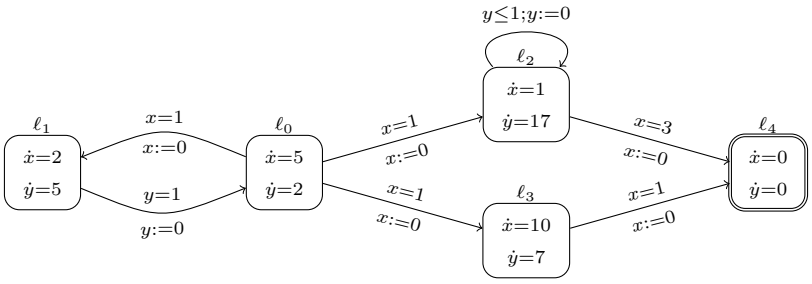

Figure 1: A singular, diagonal-free, multirate hybrid automaton.

Assume we want to reach location ℓ_4 from $(\ell_0, 0, 0)$ within one time unit. One clearly see that the duration of any run starting from ℓ_0 and crossing ℓ_2 will exceed one time unit. An other possibility would be to directly go from ℓ_0 to ℓ_3. In this case, when reaching location ℓ_3, after crossing e_{03}, the value of the variable x (resp. y) is 0 (resp. $\frac{2}{5}$). Thus, in order to cross e_{34}, one should wait $\frac{1}{10}$ time units, if we do so,

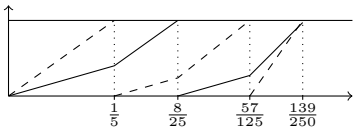

Figure 2: A successful run.

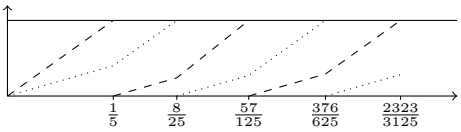

Figure 3: A loop between ℓ_0 and ℓ_1.

the value of y will reach $\frac{11}{10}$ and violate the invariant. It is thus impossible to reach ℓ_3 from $(\ell_0, 0, 0)$ without visiting ℓ_1. A single visit to ℓ_1 is sufficient as the following run testifies: $(\ell_0, 0, 0) \xrightarrow{\frac{1}{5}, e_{01}} \left(\ell_1, 0, \frac{2}{5}\right) \xrightarrow{\frac{3}{25}, e_{10}} \left(\ell_0, \frac{6}{25}, 0\right) \xrightarrow{\frac{17}{125}, e_{03}} \left(\ell_3, 0, \frac{34}{125}\right) \xrightarrow{\frac{1}{10}, e_{34}} \left(\ell_4, 0, \frac{243}{250}\right)$. The illustration of the evolution of the variables along this run is given in Fig. 2. In this picture, the evolution of the x-variable (resp. of the y-variable) is represented by the dashed (resp. plain) curve. The evolutions of the valuations of the variables along the beginning of the unique run looping between ℓ_0 and ℓ_1 is illustrated in Fig. 3. Looking at that looping run, one could be convinced that $\mathcal{H}$ does not admit a finite bisimulation quotient.

3 Decidability for RHA with Non-Negative Rates

In this section, we prove that the time-bounded reachability problem is *decidable* for the class of (non-initialized) *rectangular* hybrid automata having *non-negative rates*, while it is *undecidable* for this class in the classical (unbounded) case [7]. Note that this class is interesting in practice since it contains, among others, the important class of *stopwatch automata*, a significant subset of LHA that has several useful applications [3]. We obtain decidability by showing that for RHA with non-negative rates, a goal location is reachable within $\mathbf{T}$ time units iff there exists a witness run of that automaton which reaches the goal (within $\mathbf{T}$ time units) by a run ρ of length $|\rho| \leq K_{\mathbf{T}}^{\mathcal{H}}$ where $K_{\mathbf{T}}^{\mathcal{H}}$ is a parameter that depends on $\mathbf{T}$ and on the size of the automaton $\mathcal{H}$. Time-bounded reachability can thus be reduced to the satisfiability of a formula in the first order theory of the reals encoding the existence of runs of length at most $K_{\mathbf{T}}^{\mathcal{H}}$ and reaching Goal.

For simplicity of the proofs, we consider RHA with the following restrictions: (i) the guards *do not contain strict inequalities*, and (ii) the rates are *singular*. We argue at the end of this section that these restrictions can be made without loss of generality. Then, in order to further simplify the presentation, we show how to syntactically simplify the automaton while preserving the time-bounded reachability properties. The details of the constructions can be found in the appendix.

Proposition 1 *Let $\mathcal{H}$ be a singular RHA with non-negative rates and without strict inequalities, and let* Goal *be a location of $\mathcal{H}$. We can build a hybrid automaton $\mathcal{H}'$ with the following the properties:*

- H_1 *$\mathcal{H}'$ is a singular RHA with non-negative rates*
- H_2 *$\mathcal{H}'$ contains only deterministic resets*

H_3 *for every edge* (ℓ, g, r, ℓ') *of* $\mathcal{H}'$*,* g *is either* $\mathbf{true}$ *or of the form* $x_1 = 1 \wedge x_2 = 1 \wedge \cdots \wedge x_k = 1$*, and* $r \equiv x'_1 = 0 \wedge \cdots \wedge x'_k = 0$*.*

and a set of locations S *of* $\mathcal{H}'$ *such that* $\mathcal{H}$ *admits a* $\mathbf{T}$*-time bounded run reaching* Goal *iff* $\mathcal{H}'$ *admits a strict* 1*-variable-bounded, and* $\mathbf{T}$*-time bounded run reaching* S*.*

Proof. The proof is given in Appendix A □ As a consequence, to prove decidability of time-bounded reachability of RHA with non-negative rates, we only need to prove that we can decide whether an RHA respecting H_1 through H_3 admits a *strict* run ρ reaching the goal within $\mathbf{T}$ time units, and where all variables are bounded by 1 along ρ.

Bounding the number of equalities. As a first step to obtain a witness of time-bounded reachability, we bound the number of transitions guarded by equalities along a run of bounded duration:

Proposition 2 *Let* $\mathcal{H}$ *be an LHA, with set of variables* X *and respecting hypothesis* H_1 *through* H_3*. Let* ρ *be a* $\mathbf{T}$*-time bounded run of* $\mathcal{H}$*. Then,* ρ *contains at most* $|X| \cdot \mathrm{rmax} \cdot \mathbf{T}$ *transitions guarded by an equality.*

Proof. For a contradiction, assume that there exists an execution ρ of $\mathcal{H}$ with M transitions containing (at least) an equality where $M > |X| \cdot \mathrm{rmax} \cdot \mathbf{T}$. By H_3, the equalities in the guards are of the form $x = 1$. In particular, there must exists a variable $y \in X$ which has been tested equal to one (and thus reset to zero by H_3) strictly more than $\mathrm{rmax} \cdot \mathbf{T}$ times. Since all the rates of y are non negative by H_1, the shortest time needed to reach the guard $y = 1$ from the value 0 is $\frac{1}{\mathrm{rmax}}$. Along ρ, the variable y has reached the guard $y = 1$ from 0 strictly more than $\mathrm{rmax} \cdot \mathbf{T}$ times; this implies that $\mathsf{duration}\,(\rho) > \mathrm{rmax} \cdot \mathbf{T} \cdot \frac{1}{\mathrm{rmax}} = \mathbf{T}$ which is a contradiction. □

Bounding runs without equalities. Unfortunately, it is not possible to bound the number of transitions that do not contain equalities, even along a time-bounded run. However, we will show that, given a time-bounded run ρ without equality guards, we can build a run ρ' that is equivalent to ρ (in a sense that its initial and target states are the same), and whose length is *bounded* by a parameter depending on the size of the automaton. More precisely:

Proposition 3 *Let* $\mathcal{H}$ *be an RHA with non-negative rates. For any* 1*-variable bounded and* $\frac{1}{\mathrm{rmax}+1}$*-time bounded run* $\rho = s_0 \xrightarrow{\pi} s$ *of* $\mathcal{H}$ *that contains no equalities in the guards,* $\mathcal{H}$ *admits a* 1*-variable bounded and* $\frac{1}{\mathrm{rmax}+1}$*-time bounded run* $\rho' = s_0 \xrightarrow{\pi'} s$ *such that* $|\rho'| \leq 2|X| + (2|X| + 1) \cdot |\mathrm{Loc}| \cdot (2^{(|\mathrm{Edges}|+1)} + 1)$*.*

Note that Proposition 3 applies only to runs of duration at most $\frac{1}{\mathrm{rmax}+1}$. However, this is not restrictive, since any $\mathbf{T}$-time-bounded run can always be split into at most $\mathbf{T} \cdot (\mathrm{rmax} + 1)$ subruns of duration at most $\frac{1}{\mathrm{rmax}+1}$, provided that we add a self-loop with guard $\mathbf{true}$ and no reset on every location (this can be done without loss of generality as far as reachability is concerned).

To prove Proposition 3, we rely on a *contraction operation* that receives a *timed path* and returns another one of smaller length. Let $\pi = (t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$ be a timed path. We define $\mathsf{Cnt}(\pi)$ by considering two cases. Let j, k, j', k' be four positions such that $1 \leq j \leq k < j' \leq k' \leq n$ and $e_j \ldots e_k = e'_j \ldots e'_k$ is a *simple cycle*. **If** such j, k, j', k' exist, then let:

$$\mathsf{Cnt}(\pi) \quad = \quad \pi[1 : j-1] \cdot (e_j, t_j + t_{j'}) \cdots (e_k, t_k + t_{k'}) \cdot \pi[k+1 : j'-1] \cdot \pi[k'+1 : n]$$

Otherwise, we let $\mathsf{Cnt}(\pi) = \pi$. Observe that π and $\mathsf{Cnt}(\pi)$ share the same source and target locations, even when $\pi[k'+1 : n]$ is empty.

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 $\mathsf{Cnt}^*(\pi) = \mathsf{Cnt}^n(\pi)$ where n is the least value such that $\mathsf{Cnt}^n(\pi) = \mathsf{Cnt}^{n+1}(\pi)$. Clearly, since π is finite, and since $|\mathsf{Cnt}(\pi)| < |\pi|$ or $\mathsf{Cnt}(\pi) = \pi$ for any π, $\mathsf{Cnt}^*(\pi)$ always exists. Moreover, we can always bound the length of $\mathsf{Cnt}^*(\pi)$. This stems from the fact that $\mathsf{Cnt}^*(\pi)$ is a timed path that contains at most one occurrence of each simple cycle. The length of such paths can be bounded using classical combinatorial arguments.

Lemma 1 *For any timed path π of an LHA $\mathcal{H}$ with $|\mathrm{Loc}|$ locations and $|\mathrm{Edges}|$ edges:* $|\mathsf{Cnt}^*(\pi)| \leq |\mathrm{Loc}| \cdot (2^{(|\mathrm{Edges}|+1)} + 1)$.

Proof. Let $\mathsf{Cnt}^*(\pi) = (t_1, e_1), (t_2, e_2), \ldots, (t_n, e_n)$. First, observe that, by definition of Cnt^*, the actual values of the time delays t_1, t_2,$\ldots$ t_n are irrelevant to the length of $\mathsf{Cnt}^*(\pi)$, since the 'contraction' is based solely on the edges. Still by definition of Cnt^*, also observe that the path $e_1, e_2, \ldots, e_n$ does not contain two occurrences of the same simple cycle. Thus, the length of $\mathsf{Cnt}^*(\pi)$ is always bounded by the length of the maximal path in $\mathcal{H}$ that does not contain two occurrences of the same simple cycle.

In order to compute this value, we first observe that any path $\sigma = e_1, e_2, \ldots e_n$ can always be decomposed into subpaths $\sigma_1, \sigma_2, \ldots \sigma_{2k}, \sigma_{2k+1}$ where any σ_{2i+1} (for $0 \leq i \leq k$) is an acyclic path and any σ_{2j} is a simple cycle (for $1 \leq j \leq k$). This stems from the fact that any cycle (whether it is simple or not) can always be decomposed into a sequence of simple cycles and acyclic paths.

Thus, the worst case scenario for a path containing at most one each simple cycle is to have a path of the form: $\sigma_1, \sigma_2, \ldots \sigma_{2k}, \sigma_{2k+1}$ where each σ_{2i+1} (for $0 \leq i \leq k$) is of maximal length, and $\{\sigma_{2j} \mid 1 \leq j \leq k\}$ is the set of all possible simple cycles. By definition of a simple cycle, in an automaton with $|\mathrm{Edges}|$ and $|\mathrm{Loc}|$ locations, there are at most $2^{|\mathrm{Edges}|}$ simple cycles, and each of them has at most length $|\mathrm{Loc}|$ (otherwise the cycle would contain two edges with the some origin and the cycle wouldn't be simple). Moreover, in such an automaton, each acyclic path is of length at most $|\mathrm{Loc}|$ too. Hence, the worst case is a path $\sigma_1, \sigma_2, \ldots \sigma_{2k}, \sigma_{2k+1}$ where, $k = 2^{|\mathrm{Edges}|}$, for all $1 \leq i \leq k$: $|\sigma_{2i}| = |\mathrm{Loc}|$ and for all $0 \leq j \leq k$: $|\sigma_{2j+1}| = |\mathrm{Loc}|$, that is a total length of $k \cdot |\mathrm{Loc}| + (k+1) \cdot |\mathrm{Loc}| = |\mathrm{Loc}| \cdot (2k+1) = |\mathrm{Loc}| \cdot (2^{(|\mathrm{Edges}|+1)} + 1)$. □

Note that the contraction operation is purely syntactic and works on the timed path only. Hence, given a run $s_0 \xrightarrow{\pi} s$, we have no guarantee that $\mathsf{Run}(s_0, \mathsf{Cnt}^*(\pi)) \neq \bot$. Moreover, even in the alternative, the resulting run might be $s_0 \xrightarrow{\mathsf{Cnt}^*(\pi)} s'$ with $s \neq s'$. Nevertheless, we can show that $\mathsf{Cnt}^*(\pi)$ preserves some properties of π. For

a timed path $\pi = (t_1, e_1), \ldots, (t_n, e_n)$ of an LHA $\mathcal{H}$ with rate function Rates, we let $\mathsf{Effect}(\pi, x) = \sum_{i=1}^{n} \mathrm{Rates}(\ell_i)(x) \cdot t_i$, where ℓ_i is the initial location of e_i for any $1 \leq i \leq n$. Note thus that, for any run $(\ell, \nu) \xrightarrow{\pi} (\ell', \nu')$, for any variable x *which is not reset along* π, $\nu'(x) = \nu(x) + \mathsf{Effect}(\pi, x)$. It is easy to see that $\mathsf{Cnt}^*(\pi)$ preserves the effect of π. Moreover, the duration of $\mathsf{Cnt}^*(\pi)$ and π are equal.

Lemma 2 *For any timed path* π*:* (i) $\mathsf{duration}(\pi) = \mathsf{duration}(\mathsf{Cnt}^*(\pi))$ *and* (ii) *for any variable* x*:* $\mathsf{Effect}(\pi, x) = \mathsf{Effect}(\mathsf{Cnt}^*(\pi), x)$.

We are now ready to show, given a timed path π (with $\mathsf{duration}(\pi) \leq \frac{1}{\mathrm{rmax}+1}$ and without equality tests in the guards), how to build a timed path $\mathsf{Contraction}(\pi)$ that fully preserves the values of the variable, as stated in Proposition 3. The key ingredient to obtain $\mathsf{Contraction}(\pi)$ is to apply Cnt^* to selected portions of π, in such a way that for each edge e that resets a variable for the *first* or the *last* time along π, the time distance between the occurrence of e and the beginning of the timed path is the same in both π and $\mathsf{Contraction}(\pi)$.

The precise construction goes as follows. Let $\pi = (t_1, e_1), \ldots, (t_n, e_n)$ be a timed path. For each variable x, we denote by S_x^π the set of positions i such that e_i is either the *first* or the *last* edge in π to reset x (hence $|S_x^\pi| \in \{0, 1, 2\}$ for any x). Then, we decompose π as: $\pi_1 \cdot (t_{i_1}, e_{i_1}) \cdot \pi_2 \cdot (t_{i_2}, e_{i_2}) \cdots (t_{i_k}, e_{i_k}) \cdot \pi_{k+1}$ with $\{i_1, \ldots, i_k\} = \cup_x S_x^\pi$. From this decomposition of π, we let $\mathsf{Contraction}(\pi) = \mathsf{Cnt}^*(\pi_1) \cdot (t_{i_1}, e_{i_1}) \cdot \mathsf{Cnt}^*(\pi_2) \cdot (t_{i_2}, e_{i_2}) \cdots (t_{i_k}, e_{i_k}) \cdot \mathsf{Cnt}^*(\pi_{k+1})$.

We first note that, thanks to Lemma 1, $|\mathsf{Contraction}(\pi)|$ is bounded.

Lemma 3 *Let* $\mathcal{H}$ *be an LHA with set of variable* X*, set of edges* Edges *and set of location* Loc*, and let* π *be a timed path of* $\mathcal{H}$*. Then* $|\mathsf{Contraction}(\pi)| \leq 2 \cdot |X| + (2 \cdot |X| + 1) \cdot |\mathrm{Loc}| \cdot (2^{(|\mathrm{Edges}|+1)} + 1)$.

Proof. The Lemma stems from the fact that $|\cup_x S_x^\pi| \leq 2 \cdot |X|$ and that, for any j: $|\mathsf{Cnt}^*(\pi_j)| \leq |\mathrm{Loc}| \cdot (2^{(|\mathrm{Edges}|+1)} + 1)$ by Lemma 1. □

In order to obtain Proposition 3, it remains to show that this construction can be used to build a run ρ' that is equivalent to the original run ρ. By Lemma 2, we know that $\mathsf{duration}(\mathsf{Cnt}^*(\pi_j)) = \mathsf{duration}(\pi_j)$ for any j. Hence, the first and last resets of each variable happen at the same time (relatively to the beginning of the timed path) in both π and $\mathsf{Contraction}(\pi)$. Intuitively, preserving the time of occurrence of the first reset (of some variable x) guarantees that x will never exceed 1 along $\mathsf{Contraction}(\pi)$, because $\mathsf{duration}(\mathsf{Contraction}(\pi)) = \mathsf{duration}(\pi) \leq \frac{1}{\mathrm{rmax}+1}$. Symmetrically, preserving the last reset of some variable x guarantees that the final value of x will be the same in both π and $\mathsf{Contraction}(\pi)$. Moreover, we know (see Lemma 2) that the contraction function also preserves the value of the variables that are not reset. Thanks to these results, we are now ready to prove Proposition 3.

Proof. [of Proposition 3] Let $\pi = \mathsf{TPath}(\rho)$ and let π' denote $\mathsf{Contraction}(\pi)$. To prove the existence of ρ', we will choose $\rho' = s_0 \xrightarrow{\pi'} s$. Let us first show that $\mathsf{Run}(s_0, \pi') \neq \bot$. Since π and π' contain no equality test, by H_3, this amounts to showing that firing π' from s_0 will always keep all the variable values ≤ 1.

Let us consider the decomposition of π into: $\pi_1 \cdot (t_{i_1}, e_{i_1}) \cdot \pi_2 \cdot (t_{i_2}, e_{i_2}) \cdots (t_{i_k}, e_{i_k}) \cdot \pi_{k+1}$, as in the definition of $\mathsf{Contraction}$. For any $1 \leq i \leq k$, let $s_i = (\ell_i, \nu_i)$

denote the state reached by the run $s_0 \xrightarrow{\pi_1 \cdot (t_{i_1}, e_{i_1}) \cdots \pi_i} s_i$. Symmetrically, let $s'_i = (\ell_i, \nu'_i)$ denote the state reached by the run $s_0 \xrightarrow{\mathsf{Cnt}^*(\pi_1) \cdot (t_{i_1}, e_{i_1}) \cdots \mathsf{Cnt}^*(\pi_i)} s'_i$, assuming it exists. In that case, we observe that, for any variable x which is not reset along $\mathsf{Cnt}^*(\pi_1) \cdot (t_{i_1}, e_{i_1}) \cdots \mathsf{Cnt}^*(\pi_i)$, we have: $\nu_i(x) = \nu'_i(x)$, by Lemma 2.

Then, we proceed by contradiction. Let (t_j, e_j) be an element from π', let x be a variable such that $s_0 \xrightarrow{\pi'[1:j]} (\ell', \nu')$ and $\nu'(x) + \mathrm{Rates}(\ell')(x) \cdot t_{j+1} > 1$. We first observe that, once x has been reset along π', its value can never exceed 1 because $\mathsf{duration}(\pi') = \mathsf{duration}(\pi) \leq \frac{1}{\mathrm{rmax}+1}$. Hence, (t_j, e_j) must occur *before* the first reset of x along π'. We distinguish two cases:

1. In the case where (t_j, e_j) occurs in some part $\mathsf{Cnt}^*(\pi_{i_j})$ of the decomposition of π', we know that $\nu'_{i_j-1}(x) + \mathsf{Effect}\left((t_{i_j}, e_{i_j})\mathsf{Cnt}^*(\pi_{i_j}), x\right) > 1$, since x is not reset along $\mathsf{Cnt}^*(\pi_{i_j})$. However, we have:

$$\begin{aligned}
\nu_{i_j}(x) &= \nu_{i_j-1}(x) + \mathsf{Effect}\left((t_{i_j}, e_{i_j}) \cdot \pi_{i_j}, x\right) && \text{def. and } x \text{ not reset} \\
&= \nu'_{i_j-1}(x) + \mathsf{Effect}\left((t_{i_j}, e_{i_j}) \cdot \pi_{i_j}, x\right) && \text{observation above} \\
&= \nu'_{i_j-1}(x) + \mathsf{Effect}\left((t_{i_j}, e_{i_j}) \cdot \mathsf{Cnt}^*(\pi_{i_j}), x\right) && \text{Lemma 2} \\
&> 1
\end{aligned}$$

   Hence, ρ reaches a valuation where the value of x exceeds 1. Contradiction.

2. The case where $(t_j, e_j) = (t_{i_k}, e_{i_k})$ for some i_k is treated similarly and leads to the same contradiction.

Now, we are sure that $\rho' = s_0 \xrightarrow{\pi'} (\ell', \nu')$ is indeed a 1-variable bounded run. By Lemma 3, it has the adequate length. It remains to show that $\rho = s_0 \xrightarrow{\pi} (\ell, \nu)$ implies $\ell' = \ell$ and $\nu = \nu'$. The first point is true by definition of π'. For any variable x, let i_x denote the element (t_{i_x}, e_{i_x}) of π where the *last* reset of x occurs along π (and thus along π'). We observe that $\nu(x) = \mathsf{Effect}(\pi_{i_x+1} \cdot (t_{i_x+1}, e_{i_x+1}) \cdots \pi_{k+1}, x)$ and that $\nu'(x) = \mathsf{Effect}(\mathsf{Cnt}^*(\pi_{i_x+1}) \cdot (t_{i_x+1}, e_{i_x+1}) \cdots \mathsf{Cnt}^*(\pi_{k+1}), x)$ since x is not reset anymore along those two suffixes. By Lemma 2, we have $\nu(x) = \nu'(x)$. □

Handling '$<$' and non-singular rates. Let us now briefly explain how we can adapt the construction of this section to cope with strict guards and non-singular rates. First, when the RHA $\mathcal{H}$ contains strict guards, the RHA $\mathcal{H}'$ of Proposition 1 will also contain guards with atoms of the form $x < 1$. Thus, when building a 'contracted path' ρ' starting from a path ρ (as in the proof of Proposition 3), we need to ensure that these strict guards will also be satisfied along ρ'. It is easy to use similar arguments to establish this: if some guard $x < 1$ is not satisfied in ρ', this is necessarily before the first reset of x, which means that the guard was not satisfied in ρ either. On the other hand, to take non-singular rates into account, we need to adapt the definition of timed path. A timed path is now of the form $(t_0, r_0, e_0) \cdots (t_n, r_n, e_n)$, where each r_i is a vector of reals of size $|X|$, indicating the actual rate that was chosen for each variable when the i-th continuous step has been taken. It is then straightforward to adapt the

definitions of Cnt, Effect and Contraction to take those rates into account and still keep the properties stated in Lemma 1 and 3 and in Proposition 3 (note that we need to rely on the convexity of the invariants in RHA to ensure that proper rates can be found when building $\mathsf{Cnt}(\pi)$).

Theorem 1 *The time-bounded reachability problem is decidable for the class of rectangular hybrid automata with non-negative rates.*

Proof. Let $\mathcal{H}$ be an RHA with non-negative rates, let Goal be one of its location, let $\mathbf{B}$ be a natural value, and let us show how to determine whether $\mathcal{H}$ admits a $\mathbf{B}$-time-bounded run reaching Goal. By Proposition 1 (and taking into account the above remarks to cope with strict guards and rectangular rates), this amounts to determining the exists of a strict 1-variable bounded run reaching Goal' in $\mathcal{H}'$ (where Goal' and $\mathcal{H}'$ are defined as in Proposition 1). By Proposition 3, this can be done by considering only the runs of length at most $2|X| + (2|X|+1) \cdot |\mathrm{Loc}| \cdot (2^{(|\mathrm{Edges}|+1)} + 1)$ in $\mathcal{H}'$. This question can be answered by building an $\mathsf{FO}(\mathbb{R}, \leq, +)$ formula $\varphi_{\mathcal{H}'}$ which is satisfiable iff ρ' exists. Since the satisfiability of $\mathsf{FO}(\mathbb{R}, \leq, +)$ is decidable [4], we obtain the theorem. □

4 Undecidability Results

In this section, we show that the time-bounded reachability problem for linear hybrid automata becomes undecidable if either both positive and negative rates are allowed, or diagonal constraints are allowed in the guards. Along with the decidability result of Section 3, these facts imply that the class of rectangular hybrid automata having positive rates only and no diagonal constraints forms a maximal decidable class. Our proofs rely on reductions from the halting problem for Minsky two-counters machines.

A *two-counter machine* M consists of a finite set of control states Q, an initial state $q_I \in Q$, a final state $q_F \in Q$, a set C of counters ($|C| = 2$) and a finite set δ_M of instructions manipulating two integer-valued counters. Instructions are of the form:

$q:\ c := c+1$ **goto** q', or

$q:$ **if** $c = 0$ **then goto** q' **else** $c := c-1$ **goto** q''.

Formally, instructions are tuples (q, α, c, q') where $q, q' \in Q$ are source and target states respectively, the action $\alpha \in \{inc, dec, 0?\}$ applies to the counter $c \in C$.

A *configuration* of M is a pair (q, v) where $q \in Q$ and $v : C \to \mathbb{N}$ is a valuation of the counters. An *accepting run* of M is a finite sequence $\pi = (q_0, v_0)\delta_0(q_1, v_1)\delta_1 \ldots \delta_{n-1}(q_n, v_n)$ where $\delta_i = (q_i, \alpha_i, c_i, q_{i+1}) \in \delta_M$ are instructions and (q_i, v_i) are configurations of M such that $q_0 = q_I$, $v_0(c) = 0$ for all $c \in C$, $q_n = q_F$, and for all $0 \leq i < n$, we have $v_{i+1}(c) = v_i(c)$ for $c \neq c_i$, and (i) if $\alpha = inc$, then $v_{i+1}(c_i) = v_i(c_i) + 1$, (ii) if $\alpha = dec$, then $v_i(c_i) \neq 0$ and $v_{i+1}(c_i) = v_i(c_i) - 1$, and (iii) if $\alpha = 0?$, then $v_{i+1}(c_i) = v_i(c_i) = 0$. The *halting problem* asks, given a two-counter machine M, whether M has an accepting run. This problem is undecidable [9].

Undecidability for RHA with negative rates. Given a two-counter machine M, we construct an RHA $\mathcal{H}_M$ (thus without diagonal constraints) such that M has an accepting run if and only if the answer to the time-bounded reachability problem for $(\mathcal{H}_M, \mathrm{Goal})$ with time bound 1 is YES. The construction of $\mathcal{H}_M$ crucially makes use of both positive and negative rates.

Theorem 2 *The time-bounded reachability problem is undecidable for rectangular hybrid automata even if restricted to singular rates.*

Proof. The reduction is as follows. The execution steps of M are simulated in $\mathcal{H}_M$ by a (possibly infinite) sequence of *ticks* within one time unit. The ticks occur at time $t_0 = 0, t_1 = 1 - \frac{1}{4}, t_2 = 1 - \frac{1}{16}, \ldots$ The counters are encoded as follows. If the value of counter $c \in C$ after i execution steps of M is $v(c)$, then the variable x_c in $\mathcal{H}_M$ has value $\frac{1}{4^{i+v(c)}}$ at time t_i. Note that this encoding is time-dependent and that the value of x_c at time t_i is always smaller than $1 - t_i = \frac{1}{4^i}$, and equal to $\frac{1}{4^i}$ if the counter value is 0. To maintain this encoding (if a counter c is not modified in an execution step), we need to divide x_c by 4 before the next tick occurs. We use the divisor gadget in Figure 4 to do this. Using the diagram in the figure, it is easy to check that the value of variable x_c is divided by k^2 where k is a constant used to define the variable rates. In the sequel, we use $k = 2$ and $k = 4$ (i.e., division by 4 and by 16 respectively). Note also that the division of $\nu(x_c)$ by k^2 takes $\nu(x_c) \cdot (\frac{1}{k} + \frac{1}{k^2})$ time units, which is less than $\frac{3 \cdot \nu(x_c)}{4}$ for $k \geq 2$. Since $\nu(x_c) \leq \frac{1}{4^i}$ at step t_i, the duration of the division is at most $\frac{3}{4^i} = t_{i+1} - t_i$, the duration of the next tick.

We also use the divisor gadget on a variable x_t to construct an automaton $\mathcal{A}_{\mathsf{tick}}$ that generates the ticks, as in Figure 5. We take $k = 2$ and we connect and merge the incoming and outgoing transition of the divisor gadget. Initially, we require $x_t = 1$. Since division of x_t by $k^2 = 4$ takes $\nu(x_t) \cdot (\frac{1}{k} + \frac{1}{k^2}) = \frac{3 \cdot \nu(x_t)}{4}$ time units, it turns out that the value of x_t is always $1 - t_i = \frac{1}{4^i}$ at time t_i. Therefore, we can produce infinitely many ticks within one time unit.

The automaton $\mathcal{H}_M$ is the product of $\mathcal{A}_{\mathsf{tick}}$ with the automaton constructed as follows. Assume the set of counters is $C = \{c, d\}$. For each state q of M, we construct a location ℓ_q with rate $\dot{x}_c = 0$ and $\dot{x}_d = 0$. For each instruction $(q, \cdot, \cdot, q')$ of M, we construct a transition from location ℓ_q to $\ell_{q'}$ through a synchronized product of division gadgets to maintain the encoding, as shown in Figure 6 and Figure 7. For example, the instruction (q, inc, c, q') is simulated by dividing x_c by $16 = 4^2$ and x_d by 4, which transforms for instance $x_c = \frac{1}{4^{i+n}}$ into $x'_c = \frac{1}{4^{i+n+2}}$. The decrement is implemented similarly. Note that the decrement of c requires division by 1 which is trivially realized by a location with rate $\dot{x}_c = 0$. Finally, the zero test is implemented as follows. A counter c has value 0 in step i if $x_c = 1 - t_i = \frac{1}{4^i}$. Therefore, it suffices to check that $x_c = x_t$ to simulate a zero test. To avoid diagonal constraints, we replace $x_c = x_t$ by a test $x_t = 0$ on the transition guarded by $x_c = 0$ in the divisor gadget for x_c (as suggested in Figure 7).

The set $\mathrm{Goal} = \{\ell_{q_F}\}$ contains the location corresponding to the final state q_F in M. By the above arguments, there is a one-to-one mapping between the execution of M and the run of $\mathcal{H}_M$. In particular, the counter values at step i are correctly

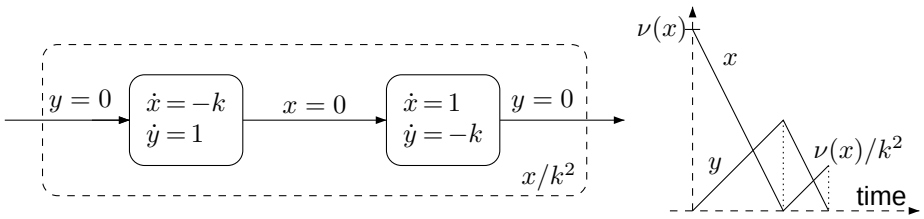

Figure 4: Gadget for division of a variable x by k^2. The variable y is internal to the gadget. The duration of the division is $v \cdot (\frac{1}{k} + \frac{1}{k^2})$. The guard $(x_t = 0)$ has no influence here, and it is used only when $k = 2$.

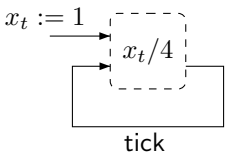

Figure 5: Tick-gadget to produce infinitely many ticks within one time unit.

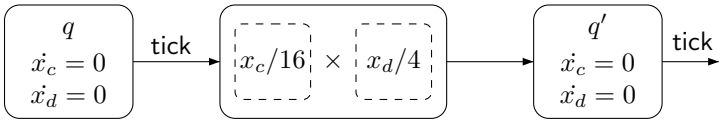

Figure 6: Increment-gadget to simulate instruction (q, inc, c, q').

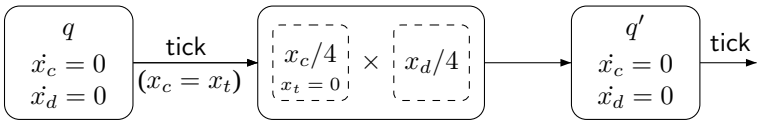

Figure 7: Zero-gadget to simulate instruction $(q, ?0, c, q')$. We do use the guard $x_t = 0$ in the divisor gadget for x_c, in order to simulate the diagonal guard $(x_c = x_t)$.

encoded at time t_i. Therefore, the location l_{q_F} is reachable in $\mathcal{H}_M$ within one time unit if and only if M has an accepting run reaching q_F. □

Undecidability with diagonal constraints. We now show that diagonal constraints also leads to undecidability. The result holds even if every variable has a positive,

singular, fixed rate.

Theorem 3 *The time-bounded reachability problem is undecidable for LHA that use only singular, strictly positive, and fixed-rate variables.*

Proof. The proof is again by reduction from the halting problem for two-counter machines. We describe the encoding of the counters and the simulation of the instructions.

Given a counter c, we represent c via two auxiliary counters c_{bot} and c_{top} such that $v(c) = v(c_{\mathrm{top}}) - v(c_{\mathrm{bot}})$.

Incrementing and decrementing c are achieved by incrementing either c_{top} or c_{bot}. Zero-testing for c corresponds to checking whether the two auxiliary counters have the same value. Therefore, we do not need to simulate decrementation of a counter.

We encode the value of counter c_{bot} using two real-valued variables x and y, by postulating that $|x - y| = \frac{1}{2^{v(c_{\mathrm{bot}})}}$. Both x and y have rate $\dot{x} = \dot{y} = 1$ at all times and in all locations of the hybrid automaton. Incrementing c_{bot} now simply corresponds to halving the value of $|x - y|$. In order to achieve this, we use two real-valued variables z and w with rate $\dot{z} = 2$ and $\dot{w} = 3$.

All operations are simulated in 'rounds'. At the beginning of a round, we require that the variables x, y, z, w have respective value $\frac{1}{2^{v(c_{\mathrm{bot}})}}, 0, 0, 0$. We first explain how we merely *maintain* the value of c_{bot} throughout a round:

1. Starting from the beginning of the round, let all variables evolve until $x = z$, which we detect via a diagonal constraint. Recall that z evolves at twice the rate of x.
2. At that point, $x = \frac{2}{2^{v(c_{\mathrm{bot}})}}$ and $y = \frac{1}{2^{v(c_{\mathrm{bot}})}}$. Reset x and z to zero.
3. Now let all variables evolve until $y = z$, and reset y, z and w to zero. It is easy to see that all variables now have exactly the same values as they had at the beginning of the round. Moreover, the invariant $|x - y| = \frac{1}{2^{v(c_{\mathrm{bot}})}}$ is maintained throughout.

Note that the total duration of the above round is $\frac{2}{2^{v(c_{\mathrm{bot}})}}$. To *increment* c_{bot}, we proceed as follows:

1′. Starting from the beginning of the round, let all variables evolve until $x = w$. Recall that the rate of w is three times that of x.

2′. At that point, $x = \frac{1.5}{2^{v(c_{\mathrm{bot}})}}$ and $y = \frac{0.5}{2^{v(c_{\mathrm{bot}})}} = \frac{1}{2^{v(c_{\mathrm{bot}})+1}}$. Reset x, z, and w to zero.

3′. Now let all variables evolve until $y = z$, and reset y, z and w to zero. We now have $x = \frac{1}{2^{v(c_{\mathrm{bot}})+1}}$, and thus the value of $|x - y|$ has indeed been halved as required.

Note that the total duration of this incrementation round is $\frac{1}{2^{v(c_{\mathrm{bot}})}}$, where $v(c_{\mathrm{bot}})$ denotes the value of counter c_{bot} prior to incrementation.

Clearly, the same operations can be simulated for counter c_{top} (using further auxiliary real-valued variables). Note that the durations of the rounds for c_{bot} and c_{top} are

in general different—in fact $c_{\rm bot}$-rounds are never faster than $c_{\rm top}$-rounds. But because they are powers of $\frac{1}{2}$, it is always possible to synchronize them, simply by repeating maintain-rounds for $c_{\rm bot}$ until the round for $c_{\rm top}$ has completed.

Finally, zero-testing the original counter c (which corresponds to checking whether $c_{\rm bot} = c_{\rm top}$) is achieved by checking whether the corresponding variables have the same value at the very beginning of a $c_{\rm bot}$-round (since the $c_{\rm bot}$- and $c_{\rm top}$-rounds are then synchronized).

We simulate the second counter d of the machine using further auxiliary counters $d_{\rm bot}$ and $d_{\rm top}$. It is clear that the time required to simulate one instruction of a two-counter machine is exactly the duration of the slowest round. Note however that since counters $c_{\rm bot}$, $c_{\rm top}$, $d_{\rm bot}$, and $d_{\rm top}$ are never decremented, the duration of the slowest round is at most $\frac{2}{2^p}$, where p is the smallest of the initial values of $c_{\rm bot}$ and $d_{\rm bot}$. If a two-counter machine has an accepting run of length m, then the total duration of the simulation is at most $\frac{2m}{2^p}$.

In order to bound this value, it is necessary before commencing the simulation to initialize the counters $c_{\rm bot}$, $c_{\rm top}$, $d_{\rm bot}$, and $d_{\rm top}$ to a sufficiently large value, for example any number greater than $\log_2(m) + 1$. In this way, the duration of the simulation is at most 1.

Initializing the counters in this way is straightforward. Starting with zero counters (all relevant variables are zero) we repeatedly increment $c_{\rm bot}$, $c_{\rm top}$, $d_{\rm bot}$, and $d_{\rm top}$ a nondeterministic number of times, via a self-loop. When each of these counters has value k, we can increment all four counters in a single round of duration $\frac{1}{2^k}$ as explained above. So over a time period of duration at most $\sum_{k=0}^{\infty} \frac{1}{2^k} = 2$ the counters can be initialized to $\lceil \log_2(m) + 1 \rceil$.

Let us now combine these ingredients. Given a two-counter machine M, we construct a hybrid automaton $\mathcal{H}_M$ such that M has an accepting run iff $\mathcal{H}_M$ has a run of duration at most 3 that reaches the final state Goal.

$\mathcal{H}_M$ uses the real-valued variables described above to encode the counters of M. In the initialization phase, $\mathcal{H}_M$ nondeterministically assigns values to the auxiliary counters, hence guessing the length of an accepting run of M, and then proceeds with the simulation of M. This ensures a correspondence between an accepting run of M and a time-bounded run of $\mathcal{H}_M$ that reaches Goal. □

References

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *TCS*, 138(1), 1995.

[2] R. Alur and D. L. Dill. A theory of timed automata. *Th. Comp. Sci.*, 126(2):183–235, 1994.

[3] F. Cassez and K. G. Larsen. The impressive power of stopwatches. In *Proc. of CONCUR*, LNCS 1877, pages 138–152. Springer, 1877.

[4] J. Ferrante and C. Rackoff. A decision procedure for the first order theory of real addition with order. *SIAM J. Comput.*, 4(1):69–76, 1975.

[5] G. Frehse. Phaver: algorithmic verification of hybrid systems past hytech. *Int. J. Softw. Tools Technol. Transf.*, 10:263–279, May 2008.

[6] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. Hytech: A model checker for hybrid systems. In *Proc. of CAV*, LNCS 1254, pages 460–463. Springer, 1997.

[7] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? *J. Comput. Syst. Sci.*, 57(1):94–124, 1998.

[8] T. A. Henzinger and J.-F. Raskin. Robust undecidability of timed and hybrid systems. In *Proc. of HSCC*, LNCS 1790, pages 145–159. Springer, 2000.

[9] M. L. Minsky. *Computation: finite and infinite machines*. Prentice-Hall Inc., Englewood Cliffs, N.J., 1967. Prentice-Hall Series in Automatic Computation.

[10] J. Ouaknine, A. Rabinovich, and J. Worrell. Time-bounded verification. In *Proc. of CONCUR*, LNCS 5710, pages 496–510. Springer, 2009.

[11] J. Ouaknine and J. Worrell. Towards a theory of time-bounded verification. In *Proc. of ICALP (II)*, LNCS 6199, pages 22–37. Springer, 2010.

A Constructions to Prove Proposition 1

In this section, we expose three constructions that allow to prove Proposition 1. These three constructions have to be applied successively, starting from an RHA with non-negative rates:

1. The first construction allows to remove the *non-deterministic resets* while preserving time-bounded reachability.
2. The second construction allows to consider only runs where the variables are *bounded by* 1. Roughly speaking, it amounts to encode the integral parts of he variables in the locations and adapting the guards and invariants accordingly.
3. The third construction allows to consider *strict runs* only.

Throughout the section, we assume all the guards to be *reduced*, i.e.: (i) the same atom does not appear twice in the same guard, (ii) the only guard containing $\mathbf{true}$ is $\mathbf{true}$ and (iii) the only guard containing $\mathbf{false}$ is $\mathbf{false}$. Remark that any guard can always be replaced by an equivalent reduced guard. For any valuation ν, we denote by $\nu[S/0]$ the valuation s.t. for any x: $\nu[S/0](x) = v(x)$ if $x \notin S$ and $\nu[S/0](x) = 0$ otherwise.x

A.1 First construction: deterministic resets

Given an RHA $\mathcal{H}$ we show how to construct an RHA $\mathcal{H}'$ with only deterministic resets such that $\mathcal{H}$ is equivalent to $\mathcal{H}'$ with respect to reachability in the sense of Proposition 4. The idea of the construction is to replace non-deterministic resets in $\mathcal{H}$ with resets to 0 in $\mathcal{H}'$ and to compensate by suitably altering the guards of subsequent transitions in $\mathcal{H}'$.

Let $X = \{x_1, \ldots, x_n\}$ be a set of variables, $\mathcal{I}$ a set of real intervals including the singleton $\{0\}$, let g be a guard on X, and let $\rho \in \mathcal{I}^n$ be an n-tuple of intervals. (Intuitively $\rho(j)$ represents the interval in which variable x_j was last reset with $\rho(j) = \{0\}$ if x_j has not yet been reset.) Then we inductively define $\mathsf{Adapt}\,(g, \rho)$ as follows:

$$\begin{array}{rcl} \mathsf{Adapt}\,(g_1 \wedge g_2, \rho) & = & \mathsf{Adapt}\,(g_1, \rho) \wedge \mathsf{Adapt}\,(g_2, \rho) \\ \mathsf{Adapt}\,(x_j \in I, \rho) & = & x_j \in (I - \rho(j))\,. \end{array}$$

Here, given intervals $I, J \subseteq \mathbb{R}$, $I - J$ denotes the interval $\{x \mid \exists y \in I, z \in J : x + z = y\}$.

Let $\mathcal{H} = (X, \mathrm{Loc}, \mathrm{Edges}, \mathrm{Rates}', \mathrm{Inv}, \mathrm{Init})$ be a RHA. We construct a new RHA $\mathsf{DetReset}\,(\mathcal{H}) = (X, \mathrm{Loc}', \mathrm{Edges}', \mathrm{Rates}, \mathrm{Inv}', \mathrm{Init}')$ as follows. Writing $\mathcal{I}$ for the set of intervals used in variable resets in $\mathcal{H}$, we have:

1. $\mathrm{Loc}' = \mathrm{Loc} \times \mathcal{I}^{|X|}$.
2. For each $\big(\ell, g, r, \ell'\big) \in \mathrm{Edges}$ we have that $\big((\ell, \rho), g', r', (\ell', \rho')\big) \in \mathrm{Edges}'$, where $g' = \mathsf{Adapt}\,(g, \rho)$; $r'(j) = \perp$ and $\rho'(j) = \rho(j)$ if $r(j) = \perp$; $r'(j) = \{0\}$ and $\rho'(j) = r(j)$ if $r(j) \neq \perp$.

3. $\mathrm{Rates}'(\ell, \rho) = \mathrm{Rates}(\ell)$.
4. $\mathrm{Inv}'(\ell, \rho) = \mathsf{Adapt}\left(\mathrm{Inv}'(\ell), \rho\right)$.
5. $\mathrm{Init}' = \{(\ell, \mathbf{0}) \mid \ell \in \mathrm{Init}\}$, where $\mathbf{0} = (\{0\}, \ldots, \{0\})$.

Proposition 4 *Let ℓ be a location of $\mathcal{H}$. Then, $\mathcal{H}$ admits a $\mathbf{T}$-time-bounded run reaching ℓ **iff** $\mathsf{DetReset}(\mathcal{H})$ admits a $\mathbf{T}$-time-bounded run reaching some location of the form (ℓ, ρ).*

A.2 Second construction: variables bounded by 1

Next, we show, given an RHA $\mathcal{H}$ *with non-negative rates and deterministic resets*, how we can build an RHA $\mathsf{CBound}(\mathcal{H})$ with the same properties, and s.t. we can decide time-bounded reachability on $\mathcal{H}$ by considering only the runs of $\mathsf{CBound}(\mathcal{H})$ with the variables bounded by 1.

The idea of the construction is to encode the integer part of the variable values of $\mathcal{H}$ in the locations of $\mathsf{CBound}(\mathcal{H})$, and to keep the fractional part (thus, a value in $[0, 1]$) in the variable. To achieve this, locations of $\mathsf{CBound}(\mathcal{H})$ are of the form $(\ell, \mathbf{i})$, where ℓ is a location of $\mathcal{H}$, and $\mathbf{i}$ is a function that associates a value from $\{0, \ldots, \mathrm{cmax}\}$ to each variable. Intuitively, $\mathbf{i}(j)$ represents the integer part of x_j in the original run of $\mathcal{H}'$, whereas the fractional part is tracked by x_j (hence all the variables stay in the interval $[0, 1]$). For instance, the configuration $(\ell, 2.1, 3.2)$ of $\mathcal{H}$ is encoded by the configuration $((\ell, (2, 3)), 0.1, 0.2)$ of $\mathsf{CBound}(\mathcal{H})$. The transitions of $\mathsf{CBound}(\mathcal{H})$ are adapted from the transitions of $\mathcal{H}$ by modifying the guards to take into account the integer part encoded in the locations. This is achieved thanks to the Adapt function described hereunder. Finally, fresh transitions are added to $\mathsf{CBound}(\mathcal{H})$ that allow to reset variables whose value reach 1, while properly adapting the information about the integral part.

Let $X = \{x_1, \ldots, x_n\}$ be a set of variables, let g be a guard on X, and let $\mathbf{i} = (i_1, \ldots, i_n) \in \mathbb{N}^n$ be a tuple of natural values. Then, we define inductively $\mathsf{Adapt}(g, \mathbf{i})$ as follows:

$$\mathsf{Adapt}(x_j \leq k, \mathbf{i}) = \begin{cases} \mathbf{false} & \text{if } k < i_j \\ x_j = 0 & \text{if } k = i_j \\ \mathbf{true} & \text{if } k > i_j \end{cases};$$

$$\mathsf{Adapt}(x_j < k, \mathbf{i}) = \begin{cases} \mathbf{false} & \text{if } k \leq i_j \\ x_j < 1 & \text{if } k = i_j + 1 \\ \mathbf{true} & \text{if } k > i_j + 1 \end{cases};$$

$$\mathsf{Adapt}(x_j = k, \mathbf{i}) = \begin{cases} \mathbf{false} & \text{if } k < i_j \\ x_j = 0 & \text{if } k = i_j \\ \mathbf{false} & \text{if } k > i_j \end{cases};$$

$$\mathsf{Adapt}\,(x_j \geq k, \mathbf{i}) = \begin{cases} \mathbf{false} & \text{if } k > i_j + 1 \\ x_j = 1 & \text{if } k = i_j + 1 \\ \mathbf{true} & \text{if } k \leq i_j \end{cases};$$

$$\mathsf{Adapt}\,(x_j > k, \mathbf{i}) = \begin{cases} \mathbf{true} & \text{if } k < i_j \\ x_j > 0 & \text{if } k = i_j \\ \mathbf{false} & \text{if } k > i_j \end{cases}.$$

$\mathsf{Adapt}\,(g_1 \wedge g_2, \mathbf{i}) = \mathsf{Adapt}\,(g_1, \mathbf{i}) \wedge \mathsf{Adapt}\,(g_2, \mathbf{i})$

Given an RHA $\mathcal{H} = (X, \mathrm{Loc}, \mathrm{Edges}, \mathrm{Rates}, \mathrm{Inv}, \mathrm{Init})$ s.t. for any $(\ell, g, r, \ell') \in \mathrm{Edges}$, for any $x \in X$: $r(x)$ is either $[0, 0]$ or $\perp$ (that is, all the resets are deterministic and to zero), we build the RHA

$$\mathsf{CBound}\,(\mathcal{H}) = (X, \mathrm{Loc}', \mathrm{Edges}', \mathrm{Rates}', \mathrm{Inv}', \mathrm{Init}')$$

as follows (where cmax is the largest constant appearing in $\mathcal{H}$):

1. $\mathrm{Loc}' = \mathrm{Loc} \times \{0, \ldots, \mathrm{cmax}\}^n$.
2. For each $\big(\ell, g, r, \ell'\big) \in \mathrm{Edges}$ we have that:

   $\big((\ell, \mathbf{i}), \mathsf{Adapt}\,(g, \mathbf{i}), r, (\ell', \mathbf{i}')\big) \in \mathrm{Edges}'$, where $i'_j = \begin{cases} i_j & \text{if } r(x_j) \neq \perp \\ 0 & \text{otherwise.} \end{cases}$

   $\big((\ell, \mathbf{i}), x_k = 1, \{x_k\}, (\ell, \mathbf{i}')\big) \in \mathrm{Edges}'$, where $i'_j = \begin{cases} i_j & \text{if } j \neq k \\ \min(i_j + 1, \mathrm{cmax}) & \text{if } j = k. \end{cases}$
3. for any $(\ell, i) \in \mathrm{Loc}'$: $\mathrm{Rates}(\ell, i) = \mathrm{Rates}(\ell)$.
4. $\mathrm{Inv}'(\ell, i) = (x_1 \leq 1) \wedge \cdots \wedge (x_n \leq 1)$, for each $(\ell, i) \in \mathrm{Loc}'$.
5. $\mathrm{Init}' = \big\{(\ell, i) \mid \ell \in \mathrm{Init}\big\}$.

Proposition 5 *Let $\mathcal{H}$ be an RHA with non-negative rates, and s.t. for any edge (ℓ, g, r, ℓ') of $\mathcal{H}$, for any variable x of $\mathcal{H}$: $r(x)$ is either $[0, 0]$ or $\perp$. Let ℓ be a location of $\mathcal{H}$. Then, $\mathcal{H}$ admits a $\mathbf{T}$-time-bounded run reaching ℓ **iff** $\mathsf{CBound}\,(\mathcal{H})$ admits a 1-variable-bounded and $\mathbf{T}$-time-bounded run reaching some location of the form $(\ell, \mathbf{i})$.*

A.3 Third construction: strictly elapsing time

Last, we explain how we can build an RHA that enforces *strictly elapsing time*. Given an RHA $\mathcal{H} = (X, \mathrm{Loc}, \mathrm{Edges}, \mathrm{Rates}, \mathrm{Inv}, \mathrm{Init})$ s.t. for any $(\ell, g, r, \ell') \in \mathrm{Edges}$, for any $x \in X$: $r(x)$ is either $[0, 0]$ or $\perp$, we build the RHA

$$\mathsf{Strict}\,(\mathcal{H}) = (X, \mathrm{Loc}', \mathrm{Edges}', \mathrm{Rates}', \mathrm{Inv}', \mathrm{Init}')$$

as follows. Let Π be the (finite) set of all non-empty paths of $\mathcal{H}$ that contains at most one occurrence of each simple loop. Then:

1. $\mathrm{Loc}' = \mathrm{Loc} \times \Pi$
2. $\big((\ell,\pi), g, r, (\ell',\pi')\big) \in \mathrm{Edges}'$ iff:
   - $\pi = (\ell, g_1, r_1, \ell_1)(\ell_1, g_2, r_2, \ell_2)\dots(\ell_{n-1}, g_n, r_n, \ell')$
   - $g = \bigwedge_{i=0}^{n} g_i[X_i/0]$, where $X_i = \{x \mid \exists 0 \leq j < i : r_j(x) \neq \bot\}$
   - r is s.t. for any $x \in X$: $r(x) = 0$ if there is $1 \leq j \leq n$ s.t. $r(j) \neq \bot$, and $r(x) = \bot$ otherwise.
3. Rates' is s.t. $\mathrm{Rates}'(\ell,\pi) = \mathrm{Rates}(\ell)$ for any $(\ell,\pi) \in \mathrm{Loc}'$.
4. Inv' is s.t.: $\mathrm{Inv}'(\ell,\pi) = \mathrm{Inv}(\ell) \wedge \bigwedge_{i=1}^{n} \mathrm{Inv}(\ell_i)[X_i/0]$ where $X_i = \{x \mid \exists 0 \leq j \leq i : r_j(x) \neq \bot\}$
5. $\mathrm{Init}' = \{(\ell,\pi) \mid \ell \in \mathrm{Init}\}$.

Proposition 6 *Let $\mathcal{H}$ be an RHA with non-negative rates and s.t. for any edge (ℓ, g, r, ℓ') of $\mathcal{H}$, for any variable x of $\mathcal{H}$: $r(x)$ is either $[0,0]$ or $\bot$. Let ℓ be a location of $\mathcal{H}$. Then, $\mathcal{H}$ admits a 1-variable-bounded and $\mathbf{T}$-time-bounded run reaching ℓ **iff** $\mathsf{Strict}(\mathcal{H})$ admits a* strict, *1-variable-bounded and $\mathbf{T}$-time-bounded run reaching some location of the form (ℓ,π).*

A.4 Proof of Proposition 1

By applying successively the three constructions above to any RHA with non-negative rates $\mathcal{H}$, one obtain an RHA $\mathcal{H}' = \mathsf{Strict}(\mathsf{CBound}(\mathsf{DetReset}(\mathcal{H})))$ that has the following properties:

1. $\mathcal{H}'$ contains only *deterministic resets* to zero
2. All the guards and invariants in $\mathcal{H}'$ are either $\mathbf{true}$ or conjunctions of atoms of the form $x = 1$ or $y < 1$ only[2]. Moreover, each time a variable is tested to 1 by an edge, it is reset to zero.

Moreover, when the original $\mathcal{H}$ contains no strict inequalities in the guards and invariants, the same holds for the guards and invariants of $\mathcal{H}'$, i.e., they will all be either $\mathbf{true}$ or of the form $x_1 = 1 \wedge x_2 = 1 \wedge \dots \wedge x_k = 1$ for $\{x_1,\dots,x_k\} \subseteq X$. Thus, $\mathcal{H}'$ has the right syntax, and respects H_1 through H_3. Given a location ℓ of $\mathcal{H}$, we let Goal bet the set of all $\mathcal{H}'$ locations of the form $(((\ell,\rho),\mathbf{i}),S)$. Thanks to Proposition 4, 5 and 6, we are ensured that $\mathcal{H}$ admits a $\mathbf{T}$-time-bounded run reaching ℓ iff $\mathcal{H}'$ admits a strict 1-variable-bounded and $\mathbf{T}$-time-bounded run reaching Goal. □

[2]Remark that the third construction removes from the guards all the atoms of the form $x > 0$ that are introduced by the second one.