Polynomial Interrupt Timed Automata: Verification and Expressiveness[☆]

B. Bérard[a], S. Haddad[b], C. Picaronny[b], M. Safey El Din[c], M. Sassolas[d]

[a] *Sorbonne Université, LIP6, CNRS UMR 7606, Paris, France*
[b] *Université Paris Saclay, École Normale Supérieure Paris Saclay, LSV, CNRS UMR 8643, INRIA Saclay IdF Center, MEXICO Team, Cachan, France*
[c] *Sorbonne Université, LIP6, CNRS UMR 7606, INRIA Paris Center, PolSys Team, Paris, France*
[d] *Université Paris-Est, LACL, Créteil, France*

Abstract

Interrupt Timed Automata (ITA) form a subclass of stopwatch automata where reachability and some variants of timed model checking are decidable even in presence of parameters. They are well suited to model and analyze real-time operating systems. Here we extend ITA with polynomial guards and updates, leading to the class of polynomial ITA (PolITA). We prove that reachability is decidable in 2EXPTIME on PolITA, using an adaptation of the *cylindrical algebraic decomposition* algorithm for the first-order theory of reals. We also obtain decidability for the model checking of a timed version of CTL and for reachability in several extensions of PolITA. In particular, compared to previous approaches, our procedure handles parameters and clocks in a unified way. We also study expressiveness questions for PolITA and show that PolITA are incomparable with stopwatch automata.

Keywords: Timed systems. Verification. Cylindrical decomposition.

[☆]This work is an extended version of [1].

Email addresses: Beatrice.Berard@lip6.fr (B. Bérard), haddad@lsv.ens-cachan.fr (S. Haddad), picaro@lsv.ens-cachan.fr (C. Picaronny), Mohab.Safey@lip6.fr (M. Safey El Din), mathieu.sassolas@u-pec.fr (M. Sassolas)

1. Introduction

Hybrid Automata. Hybrid systems [2] combine continuous evolution of variables according to flow functions (described by differential inclusions) in control nodes, and discrete jumps between these nodes, where the variables can be tested by guards and updated. This class of models is very expressive and all relevant verification questions (*e.g.* reachability) are undecidable. For the last twenty years, a large amount of research was devoted to approximation methods like in [3] or the identification of subclasses with decidable properties obtained by restricting the continuous dynamics and/or the discrete behavior of the systems. Among these classes lie the well known Timed Automata (TA) [4], where all variables are *clocks* evolving with rate 1 w.r.t. to global time, guards are comparisons of clocks with rational constants, and updates are resets. It is proved in [5] that reachability becomes undecidable when adding one stopwatch, *i.e.*, a clock whose rate is either 0 or 1 depending on the state, to timed automata. Decidability results were also obtained for larger classes (see [6, 7, 5, 8, 9, 10]), usually by building from the associated transition system (with uncountable state space) a finite abstraction preserving a specific class of properties, like reachability or those expressed by temporal logic formulas. In all these abstractions, a state is a pair composed of a control node and a polyhedron of variable values [5, 8].

Interrupt Timed Automata. The class of Interrupt Timed Automata (ITA), incomparable with TA, was introduced in [11, 12] as another subclass of hybrid automata with a (time-abstract) bisimulation providing a finite quotient, thus leading to decidability of reachability and some variants of timed model checking. In a basic n-dimensional ITA, control nodes are organized along n levels, with n stopwatches (also called clocks hereafter), one per level. At a given level, the associated clock is active, while clocks from lower levels are frozen and clocks from higher levels are irrelevant. Guards are linear constraints and the clocks can be updated by linear expressions (using only clocks from lower levels). The hierarchical structure of ITA makes them particularly well suited for modeling systems with interruptions, like real-time operating systems. ITA were extended with parameters in [13], while preserving the decidability results.

Polynomial constraints. Linear constraints are not always expressive enough for modeling purposes. In an untimed setting, polynomials of discrete variables were considered for the analysis of programs [14, 15, 16]. In the context of hybrid systems, several biological models shown in [17] involve polynomials (or even rational functions) of continuous variables, that appear in the differential equations, but can also be compared with threshold values. Polynomials of parameters were also considered in [18] for the analysis of Slope Parametric Linear Hybrid Automata (SPLHA). Classical polyhedron-based abstractions [19, 20] are not sufficient to deal with these constraints.

As a toy example to motivate the model, we consider the landing of a rocket. In the first stage, the rocket approaches the land from distance d, under gravitation g. In the second stage, the rocket approaches the land with constant deceleration $h < 0$. The rocket must reach the land with small positive speed (less than some fixed ε). The problem we are interested in is the following: For all $g \in [7, 10]$, does there exist an $h \in [-3, -1]$ such that the rocket is landing without crash ?

Contribution. We define the class PolITA, of polynomial ITA, where linear expressions are replaced by polynomials with rational coefficients both for guards and updates. For instance, a guard at level 2 with clock x_2 can be of the form $P_1(x_1)x_2^2 + P_2(x_1) \geq 0$, where P_1 and P_2 are polynomials with rational coefficients and single variable x_1, the clock of level 1.

The PolITA below illustrates the rocket landing example. The first stage is modeled by q_0 at level 1 where the system spends x_1 seconds. The the system spends x_2 seconds in q_1, representing the second stage (deceleration). State q_2 is the final state where the rocket has landed. Remark that the rocket can only land if its speed is below ε as enforced by the guard.

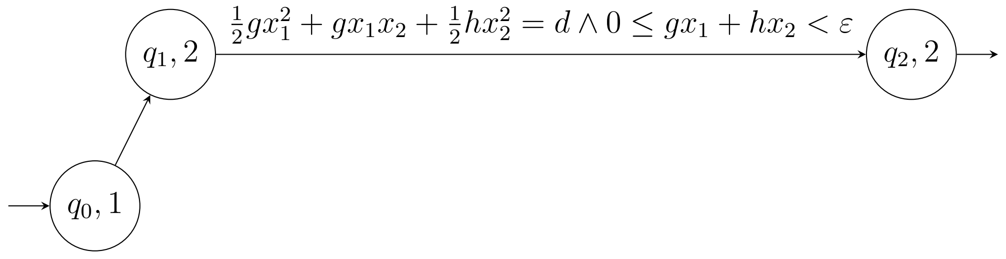

$\frac{1}{2}gx_1^2 + gx_1x_2 + \frac{1}{2}hx_2^2 = d \wedge 0 \leq gx_1 + hx_2 < \varepsilon$
$q_0, 1$
$q_1, 2$
$q_2, 2$

Figure 1: A PolITA modeling the rocket landing.

Thus, guards are more expressive in PolITA than in the whole class of linear hybrid automata. In addition, they can be used to simulate irrational (algebraic) constraints, a case that becomes undecidable in the setting of timed automata [21].

We establish that reachability is decidable in 2EXPTIME for PolITA by adapting the cylindrical algebraic decomposition algorithm due to Collins [22] (we also refer to the textbook [23] for a self-contained exposition of this algorithm) related to the first order theory of reals. This decomposition produces a finite partition of the state space, which is the basis for the construction of a finite bisimulation quotient, together with a decision procedure for the consistency of a polynomial system. The first order theory of reals has already been used in several works on hybrid automata [9, 8, 10] but it was restricted to the dynamical part, with discrete jumps that must reinitialize the variables. It was also used for parametric analysis [18] in SPLHA (but still with linear constraints on the clocks) or approximate analysis [3] of hybrid automata. Our adaptation consists in an on-the-fly construction avoiding to build the whole decomposition. The construction can also be adapted to model checking of a timed extension of CTL.

From an expressiveness point of view, w.r.t. timed language acceptance, we show that, contrary to ITA, PolITA are incomparable with stopwatch automata (SWA [20]). We also prove that the decidability results still hold with several extensions: adding auxiliary clocks and parameters, and enriching the possible updates. In particular, parametric ITA [13] can be seen as a subclass of PolITA, and the complexity of our reachability algorithm is better than the one obtained in [13], which is in 2EXPSPACE.

Outline. We describe the model of polynomial ITA in Section 2, with an example and the presentation of the verification problems. In Section 3 we informally present the cylindrical algebraic decomposition and the decision procedures for PolITA. Then in Section 4, we detail these constructions with a special focus on the data structures and algorithmic schemes. We extend the decidability result to model checking in Section 5 and discuss expressiveness and extensions in Section 6. We conclude in Section 7. Missing proofs and constructions can be found in [24].

2. Polynomial ITA

We denote respectively by $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ and $\mathbb{R}$ the sets of natural numbers, integers, rational and real numbers, with $\mathbb{R}_{\geq 0}$ for the set of non negative real numbers. Let $X = \{x_1, \ldots, x_n\}$ be a finite set of n variables called clocks. We write $\mathbb{Q}[x_1, \ldots, x_n]$ for the set of polynomials with n variables and rational coefficients.

A *polynomial constraint* is a conjunction of constraints of the form $P \bowtie 0$ where $P \in \mathbb{Q}[x_1, \dots, x_n]$ and $\bowtie \in \{<, \leq, =, \geq, >\}$, and we denote by $\mathcal{C}(X)$ the set of polynomial constraints. We also define $\mathcal{U}(X)$, the set of *polynomial updates* over X, by: $\mathcal{U}(X) = \{\wedge_{x \in X} x := P_x \mid \forall x \; P_x \in \mathbb{Q}[x_1, \dots, x_n]\}$.

A valuation for X is a mapping $v \in \mathbb{R}^X$, also identified to the n-dimensional vector $(v(x_1), \dots, v(x_n)) \in \mathbb{R}^n$. The valuation where $v(x) = 0$ for all $x \in X$ is denoted by $\mathbf{0}$. For $P \in \mathbb{Q}[x_1, \dots, x_n]$ and v a valuation, the value of P at v is $P(v) = P(v(x_1), \dots, v(x_n))$. A valuation v satisfies the constraint $P \bowtie 0$, written $v \models P \bowtie 0$, if $P(v) \bowtie 0$. The notation is extended to a combination of polynomial constraints: $v \models \varphi$ with $\varphi = \bigwedge_i P_i \bowtie_i 0$ if $v \models P_i \bowtie_i 0$ for every i.

An update of valuation v by $u = \wedge_{x \in X} x := P_x$ in $\mathcal{U}(X)$ is the valuation $v[u]$ defined by $v[u](x) = P_x(v)$ for each $x \in X$. Hence an update is atomic in the sense that all variables are assigned simultaneously. For valuation v, delay $d \in \mathbb{R}_{\geq 0}$ and $k \in \{1, \dots, n\}$, the valuation $v' = v +_k d$, corresponding to *time elapsing of d for x_k*, is defined by $v'(x_k) = v(x_k) + d$ and $v'(x) = v(x)$ for $x \neq x_k$.

Definition 1 (PolITA). A *polynomial interrupt timed automaton* (PolITA) is a tuple $\mathcal{A} = \langle \Sigma, Q, q_0, F, X, \lambda, \Delta \rangle$, where:

- Σ is a finite alphabet, with ε the empty word in Σ^*, the set of words over Σ;
- Q is a finite set of states, q_0 is the initial state, $F \subseteq Q$ is the set of final states;
- $X = \{x_1, \dots, x_n\}$ consists of n interrupt clocks;
- the mapping $\lambda : Q \to \{1, \dots, n\}$ associates with each state its level and $x_{\lambda(q)}$ is called the *active clock* in state q;
- $\Delta \subseteq Q \times \mathcal{C}(X) \times (\Sigma \cup \{\varepsilon\}) \times \mathcal{U}(X) \times Q$ is the set of transitions. Let $q \xrightarrow{\varphi, a, u} q'$ in Δ be a transition with $k = \lambda(q)$ and $k' = \lambda(q')$. The guard φ is a conjunction of constraints $P \bowtie 0$ with $P \in \mathbb{Q}[x_1, \dots, x_k]$ (P is a polynomial over clocks from levels less than or equal to k). The update u is of the form $\wedge_{i=1}^{n} x_i := C_i$ with:
  - if $k > k'$, *i.e.* the transition decreases the level, then for $1 \leq i \leq k'$, $C_i = x_i$ and for $i > k'$, $C_i = 0$;

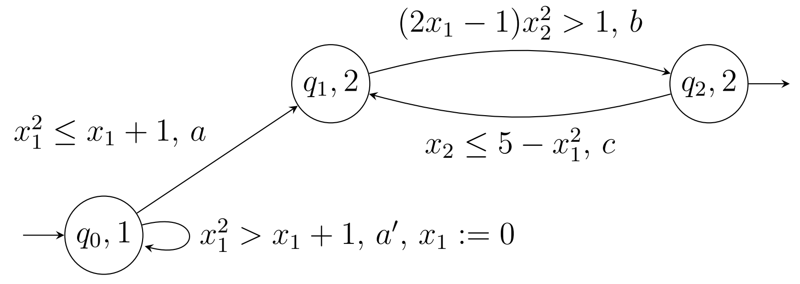


Figure 2: A sample PolITA $\mathcal{A}_0$.

– if $k \leq k'$ then for $1 \leq i < k$, $C_i = x_i$, $C_k = P$ for some $P \in \mathbb{Q}[x_1, \dots, x_{k-1}]$ or $C_k = x_k$, and for $i > k$, $C_i = 0$.

Remark that although it is possible to compare an active clock in a non-polynomial way, *e.g.* $x_2 > \sqrt{|x_1|}$ (which can be translated as $x_2^4 > x_1^2 \wedge x_2 \geq 0$), it cannot be updated in such a fashion.

Example 1. PolITA $\mathcal{A}_0$ of Figure 2 has alphabet $\{a, a', b, c\}$, two levels, with q_0 at level 1 and q_1, q_2 at level 2. The single final state is q_2. At level 1, only x_1 appears in guards and updates (here the only update is the reset of x_1 by action a'), while at level 2 guards use polynomials in both x_1 and x_2. In the sequel, the polynomials of $\mathcal{A}_0$ are denoted by $A = x_1^2 - x_1 - 1$, $B = (2x_1 - 1)x_2^2 - 1$ and $C = x_2 + x_1^2 - 5$.

A *configuration* (q, v) of $\mathcal{A}$ consists of a state q and a clock valuation v.

Definition 2. The semantics of a PolITA $\mathcal{A}$ is defined by the (timed) transition system $\mathcal{T}_\mathcal{A} = (S, s_0, \rightarrow)$, where $S = \{(q, v) \mid q \in Q,\ v \in \mathbb{R}^X\}$ is the set of configurations, with initial configuration $s_0 = (q_0, \mathbf{0})$. The relation $\rightarrow$ on S consists of two types of steps:
Time steps: Only the active clock in a state can evolve, all other clocks are frozen. For a state q with active clock $x_{\lambda(q)}$, a time step of duration $d \in \mathbb{R}_{\geq 0}$ is defined by $(q, v) \xrightarrow{d} (q, v')$ with $v' = v +_{\lambda(q)} d$.
Discrete steps: There is a discrete step $(q, v) \xrightarrow{a} (q', v')$ if there exists a transition $q \xrightarrow{\varphi, a, u} q'$ in Δ such that $v \models \varphi$ and $v' = v[u]$.

A run of a PolITA $\mathcal{A}$ is a path in the graph $\mathcal{T}_\mathcal{A}$ alternating time and discrete steps. For a given run ρ, the *trace* of ρ is the sequence of letters

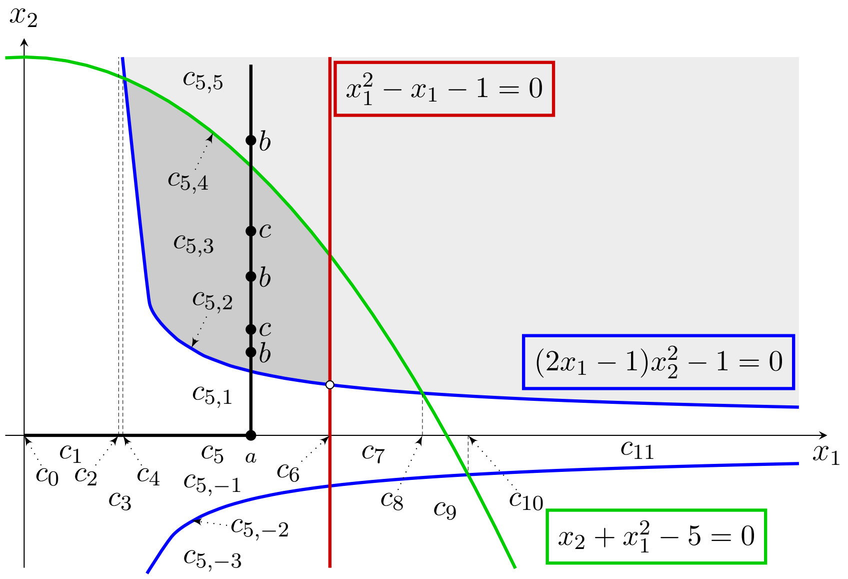


Figure 3: Sample trajectory for *abcbcb* of $\mathcal{A}_0$ in $\mathbb{R}^2$. (Axes are not orthonormal.)

(or word) appearing in the path and the *timed word* of ρ is the sequence of letters along with the absolute time of the occurrence, *i.e.* the sum of all delays appearing before the letter. A run is accepting if it ends in a state of F. The *language* (resp. *timed language*) of $\mathcal{A}$ is the set of traces (resp. timed words) of accepting runs.

Example 2. In $\mathcal{A}_0$, the transition from q_0 to q_1 can only be fired before (or when) x_1 reaches $\frac{1+\sqrt{5}}{2}$, *i.e.* at the point labeled c_6 on Figure 3. Then, transition b from q_1 to q_2 can only be taken once x_2 reaches the grey areas. Transition c cannot be taken once the green curve has been crossed. Hence the loop bc can occur as long as the clock values remain in the dark gray area $c_{5,3}$, or on the green curve $c_{5,4}$. In the sequel, we show how to symbolically compute these sets, called *cells*. Since $q_2 \in F$, the run depicted in Figure 3 is accepted by $\mathcal{A}_0$. The associated timed word (resp. trace) is $(a, 1.2)(b, 2.3)(c, 2.6)(b, 3.3)(c, 3.9)(b, 5.1)$ (resp. *abcbcb*).

Given a PoLITA $\mathcal{A}$, the *reachability problem* asks, given a state q, whether there exists a valuation v and a path from $(q_0, \mathbf{0})$ to (q, v) in $\mathcal{T}_\mathcal{A}$. Note that a variant of the reachability problem where the goal is given by both a state and polynomial constraint g on v can be treated by adding from q a transition guarded by g to a new state q_{goal} and testing reachability to q_{goal}.

The reachability procedure given in Section 3 relies on a finite abstraction of $\mathcal{T}_\mathcal{A}$. This abstraction needs to be refined enough to capture time elapsing, discrete jumps through the crossing of a transition, and keep constant the truth value of constraints $P \bowtie 0$. In the resulting model, a state will consist of an automaton state coupled with a *cell* of an appropriate *cylindrical algebraic decomposition.*

3. Cylindrical algebraic decomposition and reachability

3.1. Definition

The *cylindrical algebraic decomposition* is the basis of the first elementary decision procedure (more precisely 2EXPTIME) for the satisfiability of the first-order logic over reals [22][1]. A cylindrical decomposition of $\mathbb{R}^n$ consists of finite partitions of $\mathbb{R}, \mathbb{R}^2, \ldots, \mathbb{R}^n$ into *cells* such that the cells for $\mathbb{R}$ are open intervals or points and, at each level, cells of $\mathbb{R}^{k+1}$ are obtained by lifting each cell of $\mathbb{R}^k$ on the $k+1^{th}$ axis and then partitioning this axis with finitely many intervals and points; the partitioning itself depends on the original cell. Most of the notions presented below are introduced in [23, Chap. 5 and 11].

Example 3. Figure 3 partly depicts a cylindrical decomposition of $\mathbb{R}^2$. The cells of $\mathbb{R}_{\geq 0}$ are denoted by $c_0, \ldots, c_{11}$ (those of the negative part of the x_1 axis are not represented). They partition the line into points and intervals. For example cell c_5 is the open interval between point c_4 (corresponding to the first intersection of the blue and green curve) and $c_6 = \{\frac{1+\sqrt{5}}{2}\}$. The lifting of cell c_5 yields a partition of $c_5 \times \mathbb{R}$ into cells $c_{5,-3}, c_{5,-2}, \ldots, c_{5,5}$. Given any $z \in c_5$, $\{z\} \times \mathbb{R}$ is partitioned in an open interval $c_{5,-3} \cap \{z\} \times \mathbb{R}$ followed by a point $c_{5,-2} \cap \{z\} \times \mathbb{R}$, etc. Observe that the mapping $z \mapsto c_{5,-2} \cap \{z\} \times \mathbb{R}$ is continuous.

Definition 3. A cell of level k is a subset of $\mathbb{R}^k$ inductively defined as follows.

- When $k = 1$, it is either a point or an open interval.
- A cell C of level $k+1$ is based on a cell C' of level k. It has one of the following shapes.

[1]Later on, an EXPSPACE procedure was proposed in [25].

1. $C = \{(x, f(x)) \mid x \in C'\}$ where f is a continuous function from C' to $\mathbb{R}$;
2. $C = \{(x, y) \mid x \in C' \wedge l(x) < y < u(x)\}$ with $l < u$ continuous functions from C' to $\mathbb{R}$, possibly with $l = -\infty$ and/or $u = +\infty$.

We are interested in a cylindrical decomposition *adapted* to some finite family $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ of sets of polynomials with $\mathcal{P}_k \subseteq \mathbb{Q}[x_1, \ldots, x_k]$: in a cell of level k, the sign $(-, 0, +)$ of each polynomial in $\mathcal{P}_k$ is constant. In this context, the boundaries of each cell can be represented with algebraic constraints; hence, such cylindrical decompositions are called cylindrical algebraic decompositions. Due to the definition of cells, a cylindrical decomposition is appropriately represented by a tree.

Definition 4. A cylindrical algebraic decomposition of $\mathbb{R}^n$ adapted to $\mathcal{P} = \{\mathcal{P}_k\}_{k \leq n}$ such that $\mathcal{P}_k \subseteq \mathbb{Q}[x_1, \ldots, x_k]$, is a tree of cells inductively defined as follows:

- The root of the tree is the only cell of level 0, that is $\mathbb{R}^0$;
- Let C be a cell of level $k < n$ in the tree. There exists some $r \in \mathbb{N}$ and continuous functions f_i, for $1 \leq i \leq r$, with $-\infty = f_0 < f_1 < \ldots < f_r < f_{r+1} = +\infty$, such that the (ordered) children of C at level $k+1$ in the tree are the cells

$$\begin{array}{rcl} C_0 & = & \{(x, y) \mid x \in C \wedge f_0(x) < y < f_1(x)\}, \\ C_1 & = & \{(x, f_1(x)) \mid x \in C\}, \\ C_2 & = & \{(x, y) \mid x \in C \wedge f_1(x) < y < f_2(x)\}, \\ & \vdots & \\ C_{2r} & = & \{(x, y) \mid x \in C \wedge f_r(x) < y < f_{r+1}(x)\}, \end{array}$$

  that must satisfy: for all $P \in \mathcal{P}_{k+1}$, for all $i \in \{0, \ldots, 2r\}$, for all $z, z' \in C_i$, $sign(P(z)) = sign(P(z'))$.

Example 4. For the PoLITA of Figure 2, the relevant polynomials in $\mathbb{Q}[x_1]$ are those related to level 1: the clock x_1 itself and the polynomial $A = x_1^2 - x_1 - 1$ used in both guards from q_0, hence $\mathcal{P}_1 = \{x_1, A\}$. The relevant polynomials in $\mathbb{Q}[x_1, x_2]$ are those from level 2: x_2, $B = (2x_1 - 1)x_2^2 - 1$, and $C = x_2 + x_1^2 - 5$ associated with the guards from q_1 and q_2, so $\mathcal{P}_2 = \{x_2, B, C\}$.

We illustrate the definition above with several cells from Figure 3.

- At level 1, cells c_4, c_8, c_{10} correspond to intersection points of graphs $B = 0$ and $C = 0$ projected on the x_1 axis, while c_2 corresponds to $\frac{1}{2}$, the root of the coefficient $2x_1 - 1$ of B. Other cells like c_1, c_3, c_5 correspond to intervals between roots.
- At level 2, all cells of the form $c_{5,i}$ are children of c_5. The dark gray zone where the guards of both transitions between q_1 and q_2 are satisfied is $c_{5,3}$. It can be described by:

  $$c_{5,3} = \{(x_1, x_2) \mid x_1 \in c_5 \wedge \frac{1}{\sqrt{2x_1 - 1}} < x_2 < 5 - x_1^2\}$$

  while the next cell $c_{5,4}$ is the portion of green curve above:

  $$c_{5,4} = \{(x_1, 5 - x_1^2) \mid x_1 \in c_5\}.$$

The main elements for the effective construction of a cylindrical decomposition are given in Section 4. For the moment, we recall the result of [22]:

Theorem 1 ([22]). *For any family $\mathcal{P} = \{\mathcal{P}_k\}_{k \leq n}$ such that $\mathcal{P}_k$ is a finite subset of $\mathbb{Q}[x_1, \ldots, x_k]$, one can build a cylindrical algebraic decomposition of $\mathbb{R}^n$ adapted to $\mathcal{P}$ in 2EXPTIME, more precisely in $(|\mathcal{P}| \cdot d)^{2^{O(n)}}$ where d is the maximal degree of a polynomial of $\mathcal{P}$.*

3.2. Reachability for PolITA

We now use this decomposition to build a finite abstraction of the set of configurations of a PolITA (also called a bisimulation-quotient in the literature), which leads to the decidability of the reachability problem:

Theorem 2. *Reachability for* PolITA *is decidable in time $(d|\mathcal{A}|)^{2^{O(n)}}$ where n is the number of clocks in $\mathcal{A}$ and d the maximal degree of polynomials appearing in $\mathcal{A}$; thus in polynomial time when the number of clocks is fixed.*

Let $\mathcal{A} = \langle \Sigma, Q, q_0, F, X, \lambda, \Delta \rangle$ be a PolITA with $X = \{x_1, \ldots, x_n\}$. We define $Poly(\mathcal{A})$ as the set of all polynomials appearing in guards and updates of $\mathcal{A}$ (including all clocks) as follows: P belongs to $Poly(\mathcal{A})$ iff (1) P is a clock, (2) P occurs in a guard $P \bowtie 0$, or (3) $P = x_i - P_i$ where $x_i := P_i$ is an update.

We denote by $\mathcal{D}_\mathcal{A}$ a cylindrical decomposition adapted to $Poly(\mathcal{A})$, with $\mathcal{D}_\mathcal{A}^1, \ldots, \mathcal{D}_\mathcal{A}^n$ for the set of cells at the respective levels $1, \ldots, n$ so that for $1 \leq k \leq n$, $\mathcal{D}_\mathcal{A}^k$ is a decomposition of $\mathbb{R}^{\{x_1, \ldots, x_k\}}$.

We define a finite transition system $\mathcal{R}_\mathcal{A}$ with states in $Q \times \mathcal{D}_\mathcal{A}$. The states can also be partitioned according to levels as $\bigcup_{k=1}^{n} \lambda^{-1}(k) \times \mathcal{D}_\mathcal{A}^k$. Indeed, given a configuration (q, v) with $\lambda(q) = k$, the clocks of level $i > k$ are irrelevant and so v can be identified as a point in $\mathbb{R}^{\{x_1,\dots,x_k\}}$. We now define the transitions of $\mathcal{R}_\mathcal{A}$ as follows.

Time successors. Let *succ* $\notin \Sigma$ be a letter representing time elapsing. Let (q, C) be a state of $\mathcal{R}_\mathcal{A}$, with $\lambda(q) = k$, and let $\underline{C} \in \mathcal{D}_\mathcal{A}^{k-1}$ be the projection of C onto $\mathbb{R}^{k-1}$ and $-\infty = f_0 < \cdots < f_{r+1} = +\infty$ be the functions dividing $\underline{C}$ as in Definition 4. The *succ* transitions are defined as follows:

- if $C = \{(x, f_i(x)) \mid x \in \underline{C}\}$ for some $i \in \{1, \dots, r\}$, then there is a transition $(q, C) \xrightarrow{succ} (q, C')$ where $C' = \{(x, y) \mid x \in \underline{C}, f_i(x) < y < f_{i+1}(x)\}$;
- if $C = \{(x, y) \mid x \in \underline{C}, f_{i-1}(x) < y < f_i(x)\}$ for some $i \in \{1, \dots, r\}$, there is a transition $(q, C) \xrightarrow{succ} (q, C')$, with $C' = \{(x, f_i(x)) \mid x \in \underline{C}\}$;
- otherwise, $C = \{(x, y) \mid x \in \underline{C}, f_r(x) < y < f_{r+1}(x)\}$, and there is a self-loop labeled by *succ*: $(q, C) \xrightarrow{succ} (q, C)$.

In all the above cases, C' is called the *time successor* of C (in the last case, C is its own time successor). Since the decomposition is *cylindrical*, time elapsing according to the current clock corresponds to moving to the "next" cell.

Proposition 1 (Correctness w.r.t. time elapsing). *Let v be a valuation belonging to a cell C of level k. There exists $d > 0$ such that:*

- *The elapsing of d time units for x_k yields a valuation $v +_k d \in C'$, the time successor of C.*
- *For any $0 < d' < d$, the elapsing of d' time units for x_k yields a valuation $v +_k d$ that is either in C or in C'.*

Discrete successors. Since $\mathcal{D}_\mathcal{A}$ is adapted to *Poly*$(\mathcal{A})$ which contains all guards and updates we can write $C \models \varphi$ whenever $v \models \varphi$ for some $v \in C$ and $C[u]$ for the unique cell in $\mathcal{D}_\mathcal{A}$ such that for any valuation $v \in C$, $v[u] \in C[u]$. Discrete transitions of $\mathcal{A}$ are translated as follows into $\mathcal{R}_\mathcal{A}$. If $(q, \varphi, a, u, q') \in \Delta$, and $C \models \varphi$ with $k' = \lambda(q') < \lambda(q) = k$, the update u is trivial, hence there is a transition $(q, C) \xrightarrow{a} (q', C')$ where $C \in \mathcal{D}_\mathcal{A}^k$ has been lifted from $C' \in \mathcal{D}_\mathcal{A}^{k'}$, *i.e.* C' is the projection of C over $\mathbb{R}^{k'}$. If $(q, \varphi, a, u, q') \in \Delta$, and $C \models \varphi$ with $k' = \lambda(q') \geq \lambda(q) = k$, there is a transition $(q, C) \xrightarrow{a} (q', C')$ where $C' \in \mathcal{D}_\mathcal{A}^{k'}$ has been successively lifted $k' - k$ times from $C[u] \in \mathcal{D}_\mathcal{A}^k$ with point 0. Since the decomposition provides *sign-invariant* cells with respect to the polynomials of $\mathcal{A}$, we have:

Proposition 2 (Correctness w.r.t. discrete steps).

- *If* $(q, v) \xrightarrow{a} (q', v') \in \mathcal{T}_{\mathcal{A}}$*, then* $(q, C) \xrightarrow{a} (q', C') \in \mathcal{R}_{\mathcal{A}}$ *with* $v \in C$ *and* $v' \in C'$.
- *If* $(q, C) \xrightarrow{a} (q', C') \in \mathcal{R}_{\mathcal{A}}$ *then for all* $v \in C$ *there exists* $v' \in C'$ *such that* $(q, v) \xrightarrow{a} (q', v') \in \mathcal{T}_{\mathcal{A}}$.

Since the number of cells in a cylindrical decomposition is doubly exponential in the number of clocks and polynomial in the number and the maximal degree of polynomials to which it is adapted [23, Chap. 11], we obtain the complexity stated in Theorem 2. By setting $\{(q, C) \mid q \in F\}$ as the set of final states of $\mathcal{R}_{\mathcal{A}}$, we obtain:

Proposition 3. *The untimed language of a* PolITA *is regular.*

4. Effective construction and on-the-fly algorithm

4.1. Construction of a cylindrical decomposition

Building a cylindrical decomposition consists in two stages: the elimination stage that enlarges $\mathcal{P}$ and the lifting stage that builds the cylindrical decomposition using symbolic representations of sample points (one per cell).

Elimination stage. Starting from a cell C at level k, in order to get a partition at level $k+1$ adapted to $\mathcal{P}_{k+1}$, any two points $z, z' \in C$ should trigger a similar behavior for polynomials of $\mathcal{P}_{k+1}$, that we consider for our discussion as univariate polynomials of $\mathbb{Q}[x_1, \ldots, x_k][x_{k+1}]$ with variable x_{k+1}. More precisely, the properties we are looking for are:

- For all $P \in \mathcal{P}_{k+1}$ and for all z, z' in C, the number of real roots (counted with multiplicities) of the polynomials $P(z)$ and $P(z')$ in $\mathbb{R}[x_{k+1}]$ are equal (say μ_P). For $1 \leq i \leq \mu_P$ and $z \in C$, we denote by $r_{P,i}(z)$ the i^{th} real root of polynomial $P(z)$ (in increasing order);
- For all $P, Q \in \mathcal{P}_{k+1}$, for all $1 \leq i \leq \mu_P$ and $1 \leq j \leq \mu_Q$, for all z, z' in C, $r_{P,i}(z) \leq r_{Q,j}(z)$ implies $r_{P,i}(z') \leq r_{Q,j}(z')$.

These properties are analytical and do not provide insights on how to ensure them. Fortunately, it turns out that a simple effective semi-algebraic sufficient condition exists: there is a finite subset of polynomials of $\mathbb{Q}[x_1,\ldots,x_k]$ denoted by $Elim_{x_{k+1}}(\mathcal{P}_{k+1})$ such that if z, z' satisfy $sign(R(z)) = sign(R(z'))$ for all $R \in Elim_{x_{k+1}}(\mathcal{P}_{k+1})$, then the above properties are satisfied.

To define $Elim_{x_{k+1}}(\mathcal{P}_{k+1})$, we need some notations. For $P = \sum_{i\leq p} a_i x_{k+1}^i$ with $a_i \in \mathbb{Q}[x_1,\ldots,x_k]$ for all i, $lcof(P)$ denotes the *leading coefficient* a_p. Since this leading coefficient is a polynomial and could be null for some $P(z)$, the set of truncations of P contains all "possible realizations" of P:

$$Tru(P) = \left\{ \sum_{i\leq h} a_i x_{k+1}^i \;\middle|\; \forall i > h \ a_i \notin \mathbb{R}\setminus\{0\} \wedge a_h \neq 0 \right\}.$$

For instance, if $P = x_1x_2^3 + (3x_1+1)x_2^2 + 5x_2 - 2$, then the set of truncations of P is $Tru(P) = \{P, (3x_1+1)x_2^2 + 5x_2 - 2, 5x_2 - 2\}$. Note that even though in this case it is not possible for x_1 and $3x_1+1$ to be null at the same time, truncation is defined syntactically in order to deal with the general case.

Given another polynomial, $Q = \sum_{i\leq q} b_i x_{k+1}^i \in \mathbb{Q}[x_1,\ldots,x_k][x_{k+1}]$, the *subresultants* $(sRes_i(P,Q))_{i\leq \max(p,q)}$ are polynomials of $\mathbb{Q}[x_1,\ldots,x_k]$ obtained as determinants of matrices whose items are coefficients of P and Q (see [23, 24] for a formal definition of subresultants, a polynomial time computation, and their properties).

Definition 5. Let $\mathcal{P}_k$ be a finite subset of $\mathbb{Q}[x_1,\ldots,x_{k-1}][x_k]$ for $k > 1$. Then $Elim_{x_k}(\mathcal{P}_k)$ is the subset of $\mathbb{Q}[x_1,\ldots,x_{k-1}]$ defined for all $P, Q \in \mathcal{P}_k, R \in Tru(P)$, $T \in Tru(Q)$ by:

- if $lcof(R)$ does not belong to $\mathbb{Q}$ then $lcof(R) \in Elim_{x_k}(\mathcal{P}_k)$;
- if $deg(R) \geq 2$ then for all $sRes_j(R, \frac{\partial R}{\partial x_k})$ that are defined and do not belong to $\mathbb{Q}$, $sRes_j(R, \frac{\partial R}{\partial x_k}) \in Elim_{x_k}(\mathcal{P}_k)$;
- for all $sRes_j(R,T)$ that are defined and do not belong to $\mathbb{Q}$, we have $sRes_j(R,T) \in Elim_{x_k}(\mathcal{P})$.

Using the properties of subresultants, one gets the following theorem whose implementation is the elimination stage of the cylindrical decomposition. Due to the quadratic blow up at each recursive level of elimination, the final number of polynomials is doubly exponential w.r.t. the original number.

Theorem 3. *Let $\mathcal{P} = \{\mathcal{P}_k\}_{k \leq n}$ be a family of finite set of polynomials such that $\mathcal{P}_k \subseteq \mathbb{Q}[x_1, \ldots, x_k]$. Define $\mathcal{Q}_n = \mathcal{P}_n$ and inductively $\mathcal{Q}_{k-1} = \mathcal{P}_{k-1} \cup Elim_{x_k}(\mathcal{Q}_k)$ for $k > 1$. Then there exists a cylindrical decomposition adapted to $\mathcal{Q}$ (and thus to $\mathcal{P}$).*

Example 5. Consider again the polynomials $B = (2x_1 - 1)x_2^2 - 1$ and $C = x_2 + x_1^2 - 5$ from the PoLITA of Figure 2. Their subresultant of index 0 is $F = -2x_1^5 + x_1^4 + 20x_1^3 - 10x_1^2 - 50x_1 + 26$ which has precisely three real roots c_4, c_8, c_{10}: the x_1-coordinates of intersection points of graphs $B = 0$ and $C = 0$ mentioned previously.

Lifting stage. The starting point of the lifting stage is the family $\mathcal{P}$ appropriately enlarged by the elimination stage. In the cylindrical decomposition that we build, every cell C of level k is represented by a *sample point* inside the cell and the values of signs of all polynomials of the set $\mathcal{P}_k$ on this point.

We consider representations of real subrings of the form $\mathbb{D} = \mathbb{Q}[\alpha_1, \ldots, \alpha_k]$ where the α_i's are algebraic numbers, *i.e.*, roots of polynomials in $\mathbb{Q}[x]$. Any real algebraic number α can be represented by a pair (n, P) where P is a non null polynomial in $\mathbb{Q}[x]$ such that $P(\alpha) = 0$ and n is the index of α in the ordered set of real roots of P. This representation is extended for real algebraic points $(\alpha_1, \ldots, \alpha_k)$ with the notion of *triangular systems*: α_1 is the n_1^{th} root of $P_1 \in \mathbb{Q}[x_1]$, α_2 is the n_2^{th} root of $P_2(\alpha_1)$ with $P_2 \in \mathbb{Q}[x_1][x_2]$, etc.

Definition 6 (Triangular system). For $k \geq 1$, let $(\alpha_1, \ldots, \alpha_k)$ be a sequence of reals and let $\{(n_i, P_i)\}_{i=1}^k$ be such that for all i, n_i is a positive integer and $P_i \in \mathbb{Q}[x_1, \ldots, x_{i-1}][x_i]$. Then $\{(n_i, P_i)\}_{i=1}^k$ is a triangular system of level k for $(\alpha_1, \ldots, \alpha_k)$ if:

- P_1 is non null and α_1 is its n_1^{th} real root;
- $P_{i+1}(\alpha_1, \ldots, \alpha_i)$ is a non null polynomial of $\mathbb{Q}[\alpha_1, \ldots, \alpha_i][x_{i+1}]$, and α_{i+1} is its n_{i+1}^{th} real root, for all i, $1 \leq i < k$.

Example 6. Let us consider the point (α_1, α_2) depicted as a circle in Figure 3. This point is represented by the triangular system $((2, A), (2, B))$ where $A = x_1^2 - x_1 - 2$ and $B = (2x_1 - 1)x_2^2 - 1$. This means that α_1 is the 2^{nd} root of $A(x_1)$ and α_2 is the 2^{nd} root of $B(\alpha_1, x_2)$.

The interest of such a representation is its effectiveness: in a ring $\mathbb{D} = \mathbb{Q}[\alpha_1, \ldots, \alpha_k]$ associated with a triangular system one can compute:

1. the sign of an item of $\mathbb{Q}[\alpha_1, \ldots, \alpha_k]$,
2. the number of real roots of $P(\alpha_1, \ldots, \alpha_k)$ with $P \in \mathbb{Q}[x_1, \ldots, x_k][x_{k+1}]$,
3. the sign realizations of a polynomial $Q(\alpha_1, \ldots, \alpha_k)$ on the real roots of a polynomial $P(\alpha_1, \ldots, \alpha_k)$,

and one can order (with merge) the roots of $P(\alpha_1, \ldots, \alpha_k)$ and $Q(\alpha_1, \ldots, \alpha_k)$. All these procedures are performed in polynomial time (see for instance [24]).

The tree corresponding to the cylindrical decomposition is built top-down so that a triangular system is associated with a sample point of every cell and its sign realizations on the appropriate polynomials. Let us describe how, given a sample point $(\alpha_1, \ldots, \alpha_k)$, the partition over axis x_{k+1} can be built w.r.t. $\mathcal{P}_{k+1}$. First for all $P \in \mathcal{P}_{k+1}$, the number of roots of $P(\alpha_1, \ldots, \alpha_k)$ is determined. Then the roots of these polynomials are sorted and merged; their triangular system is the one associated with $(\alpha_1, \ldots, \alpha_k)$ extended by the polynomial for which they are roots. Then the open intervals between these roots or beyond these roots must be specified, to yield the *completed line partitioning*. Let (r, P) and (s, Q) be the borders of an open interval, then one selects as sample point, a root of $\frac{\partial(PQ)}{\partial x_{k+1}}$ located in the interval. Let (r, P) and $+\infty$ (resp. $-\infty$ and $(1, P)$) be the borders of the last (resp. first) open interval, then one selects $(r, P[x_{k+1} := x_{k+1} - 1])$ (resp. $(1, P[x_{k+1} := x_{k+1} + 1])$) as sample point. To achieve this step it remains to compute the sign realizations of $P(\alpha_1, \ldots, \alpha_k)$ for all $P \in \mathcal{P}_{k+1}$ on these sample points. Theorem 1 results from these two construction steps.

4.2. On-the-fly algorithm

The abstraction from Section 3 provides decidability of the reachability problem, by the algorithm that builds the finite graph $\mathcal{R}_{\mathcal{A}}$.

However, building the complete graph is not efficient in practice, since it requires to build the set of all cells beforehand, even though usually most of them are unreachable. Observe that the lifting stage requires to manipulate polynomials of large degrees (doubly exponential). Hence, in practice, it is often the most time consuming stage of the cylindrical algebraic decomposition algorithm. In the sequel, we show an on-the-fly construction of $\mathcal{R}_{\mathcal{A}}$ that reduces complexity in practice.

The key to the on-the-fly algorithm is to store only the part of the tree corresponding to the current sample point and its time successors. This construction relies on executing the lifting phase only when the level is increased and then only for the current sample point. As an illustration, in Figure 3, only the lifting for x_2 above c_5 has been represented, since it is the only relevant one with respect to the given trajectory. Note that liftings over sample points c_0 to c_6 have to be computed in order to build the reachable part of $\mathcal{R}_{\mathcal{A}_0}$. On the other hand, liftings over c_7 to c_{11} and over unrepresented cells to the left of c_0, need not, since level 2 is not reachable from these cells. As a result, we do not keep the whole tree but only part of it.

We show that this information is sufficient to compute the successors through time elapsing and transition firing. Although this pruning yields better performances in practice, the computational complexity in the worst case is not improved.

Definition 7 (Pruned tree). Let $\{\mathcal{P}_k\}_{k \leq n}$ be the polynomials obtained by the elimination phase. The *pruned tree* for sample point $(\alpha_1, \dots, \alpha_k)$ is the sequence of completed line partitionings for sample points $\{(\alpha_1, \dots, \alpha_i)\}_{1 \leq i \leq k}$. The pruned tree for the empty sample point ($k = 0$) is the line partitioning at level 1.

A valuation $(v_1, \dots, v_k, 0, \dots, 0)$ at level k is represented by a sample point $(\alpha_1, \dots, \alpha_k)$, or, equivalently, by a pruned tree for sample point $(\alpha_1, \dots, \alpha_{k-1})$ and the index m of α_k in the line partitioning for $(\alpha_1, \dots, \alpha_{k-1})$. In this representation, computing the time successors of $(\alpha_1, \dots, \alpha_k)$ is simply done by incrementing m (if it is not the maximal index in the line partitioning).

The set of enabled discrete transitions can be generated by computing the signs of polynomials appearing in guards. When a discrete transition $q \xrightarrow{g,a,u} q'$ is chosen, there are three cases w.r.t. the level of states q and q'.

- The level decreases, *i.e.* $\lambda(q') < \lambda(q)$. Then the pruned tree corresponding to the new configuration is the truncation of the original pruned tree up to height $\lambda(q')$. Otherwise said, we “forget” line partitionings for levels above $\lambda(q')$; however, the partitionings are kept in memory to avoid redundant computations. The new index is the index of $\alpha_{\lambda(q')}$ in the partitioned line for this level.

- The level is unchanged, *i.e.* $\lambda(q') = \lambda(q) = k$. The only possible change of clock values is through an update $x_k := P$ with $P \in \mathbb{Q}[x_1, \dots, x_{k-1}]$. The polynomial $R = x_k - P$ of degree 1 was added to $\mathit{Poly}(\mathcal{A})$ and its unique root α'_k appears in the line partitioning of level k. Note that in the triangular system representing $(\alpha_1, \dots, \alpha'_k)$ it may appear as $((n_1, P_1), \ldots, (n_k, P_k))$ with $(n_k, P_k) \neq (1, R)$. Hence to determine the index in the partitioned line the algorithm must actually determine the sign of R for all sample points of the line until 0 is found.

- The level increases, *i.e.* $\lambda(q') > \lambda(q)$. If there is an update of x_k, the same computations as above must be performed in order to find the new sample point corresponding to the valuation of clocks up to $\lambda(q)$. Then the pruned tree of height $\lambda(q')$ has to be computed (or retrieved). This is done by $\lambda(q') - \lambda(q)$ lifting steps. These lifting steps are applied on sample points of the form $(\alpha_1, \ldots, \alpha_{\lambda(q)}, 0, \ldots, 0)$, since all clocks are null for levels above $\lambda(q)$.

The on-the-fly algorithm builds the reachable part of $\mathcal{R}_{\mathcal{A}}$ as follows: the elimination phase is performed and the line for x_1 is partitioned. It starts with a queue containing q_0 with index corresponding to the root of x_1 (*i.e.* 0). Then until the queue is empty, it computes all (new) successors through time and discrete transitions, building the pruned tree as described above. As noted above, a line partitioning only needs to be computed once. In addition, and this also holds for the complete construction of $\mathcal{R}_{\mathcal{A}}$, the *triangular* structure of triangular systems enables a sharing of line partitioning at lower levels.

Example 7. We conclude this section by an example of construction of (a part of) the finite automaton $\mathcal{R}_{\mathcal{A}_0}$ associated with PolITA $\mathcal{A}_0$ (Figure 2). It is depicted on Figure 4, where points are given by the triangular system representing them. Computations of sample points for intervals between roots were omitted, and only appear in the graph as roots of derivatives. Note that having no a edge from state $q_0, 1, (5, Int)$ is not an omission, but a consequence of the guard $x_1^2 \leq x_1 + 1$ no longer being satisfied. In this graph, C_+ is the polynomial obtained when replacing x_2 by $x_2 - 1$ in C. Faded states and transitions are unreachable but are nonetheless constructed from the decomposition, though not by the on-the-fly algorithm. Gray states are those traversed by the trajectory of Figure 3. The names of the polynomials obtained by elimination and used in this graph are given in Table 1.

5. Model checking

More elaborate queries regarding the behavior of a PolITA can be expressed through temporal logics like CTL [26, 27] or timed extensions of such

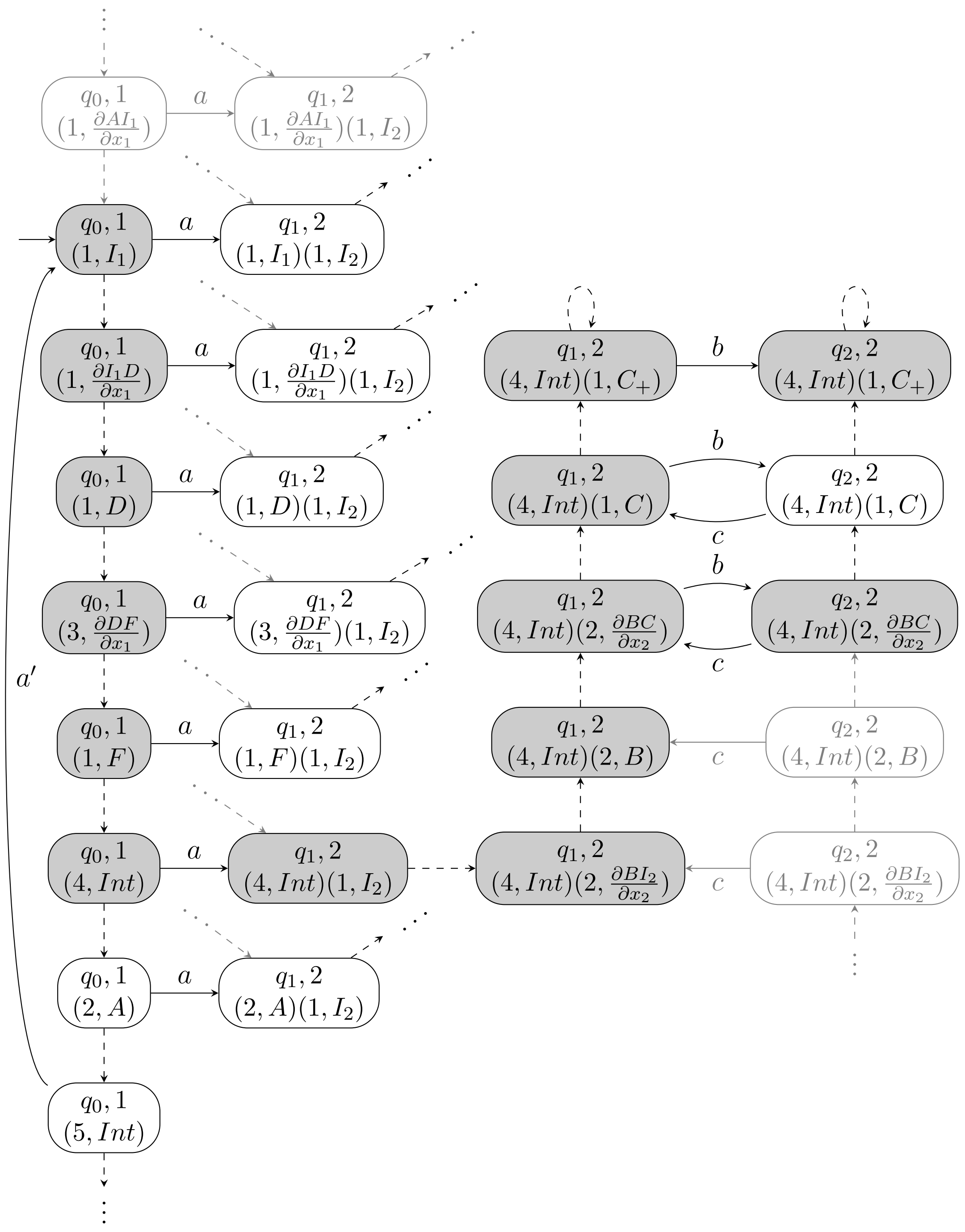

$q_0, 1$ $(1, \frac{\partial AI_1}{\partial x_1})$
a
$q_1, 2$ $(1, \frac{\partial AI_1}{\partial x_1})(1, I_2)$
$q_0, 1$ $(1, I_1)$
a
$q_1, 2$ $(1, I_1)(1, I_2)$
$q_0, 1$ $(1, \frac{\partial I_1 D}{\partial x_1})$
a
$q_1, 2$ $(1, \frac{\partial I_1 D}{\partial x_1})(1, I_2)$
$q_1, 2$ $(4, Int)(1, C_+)$
b
$q_2, 2$ $(4, Int)(1, C_+)$
$q_0, 1$ $(1, D)$
a
$q_1, 2$ $(1, D)(1, I_2)$
$q_1, 2$ $(4, Int)(1, C)$
b
c
$q_2, 2$ $(4, Int)(1, C)$
$q_0, 1$ $(3, \frac{\partial DF}{\partial x_1})$
a
$q_1, 2$ $(3, \frac{\partial DF}{\partial x_1})(1, I_2)$
$q_1, 2$ $(4, Int)(2, \frac{\partial BC}{\partial x_2})$
b
c
$q_2, 2$ $(4, Int)(2, \frac{\partial BC}{\partial x_2})$
a'
$q_0, 1$ $(1, F)$
a
$q_1, 2$ $(1, F)(1, I_2)$
$q_1, 2$ $(4, Int)(2, B)$
c
$q_2, 2$ $(4, Int)(2, B)$
$q_0, 1$ $(4, Int)$
a
$q_1, 2$ $(4, Int)(1, I_2)$
$q_1, 2$ $(4, Int)(2, \frac{\partial BI_2}{\partial x_2})$
c
$q_2, 2$ $(4, Int)(2, \frac{\partial BI_2}{\partial x_2})$
$q_0, 1$ $(2, A)$
a
$q_1, 2$ $(2, A)(1, I_2)$
$q_0, 1$ $(5, Int)$

Figure 4: Partial representation of $\mathcal{R}_{\mathcal{A}_0}$.
Dashed edges correspond to time successors *succ*; faded states are unreachable.

$$\begin{array}{rcl}
I_1 & = & x_1 \\
I_2 & = & x_2 \\
A & = & x_1^2 - x_1 - 1 \\
B & = & (2x_1 - 1)x_2^2 - 1 \\
C & = & x_2 + x_1^2 - 5 \\
D & = & 2x_1 - 1 \ (= lcof(B)) \\
E & = & x_1^2 - 5 \ (= sRes_0(I_2, C)) \\
F & = & -2x_1^5 + x_1^4 + 20x_1^3 - 10x_1^2 - 50x_1 + 26 \ (= sRes_0(B, C)) \\
G & = & 4(2x_1 - 1)^2 \ (= sRes_0(B, \frac{\partial B}{\partial x_2})) \\
Int & = & -14x_1^6 + 18x_1^5 + 105x_1^4 - 124x_1^3 - 180x_1^2 + 172x_1 + 24 \ (= \frac{\partial FA}{\partial x_1})
\end{array}$$

$$\mathcal{P}_1 = \{I_1, A, D, E, F, G\} \qquad \mathcal{P}_2 = \{I_2, B, C\}$$

Table 1: Polynomials used in the cylindrical decomposition.

logics like TCTL [19, 28]. Here we consider a timed extension of CTL where it is possible to reason on the values of clocks of the PolITA.

Given a PolITA $\mathcal{A} = \langle \Sigma, Q, q_0, F, X, \lambda, \Delta \rangle$ and a set AP of atomic propositions, we equip the automaton with a mapping $lab : Q \to 2^{AP}$, labeling a state with the set of propositions that hold in this state. For convenience, we assume that $Q \subseteq AP$ with for all $q, q' \in Q$, $q' \in lab(q)$ iff $q = q'$.

Definition 8. Formulas of the timed logic $\mathsf{TCTL}_{\mathrm{int}}$ are defined by the following grammar:

$$\psi ::= p \mid \psi \wedge \psi \mid \neg\psi \mid P \bowtie 0 \mid \mathsf{A}\, \psi\, \mathsf{U}\, \psi \mid \mathsf{E}\, \psi\, \mathsf{U}\, \psi$$

where $p \in AP$, P is a polynomial of $\mathbb{Q}[x_1, \ldots, x_n]$, and $\bowtie \in \{>, \geq, =, \leq, <\}$.

The formulas of $\mathsf{TCTL}_{\mathrm{int}}$ are interpreted over configurations of $\mathcal{A}$, hence the semantics of $\mathsf{TCTL}_{\mathrm{int}}$ is defined as follows on the transition system $\mathcal{T}_{\mathcal{A}}$ associated with $\mathcal{A}$. Let $Run(s)$ denote all runs starting from configuration $s = (q, v)$. For $\rho = (q, v) \xrightarrow{d_1} (q, v +_{\lambda(q)} d_1) \xrightarrow{a_1} (q_2, v_2) \cdots \in Run(s)$, a position in ρ is a pair $\pi = (i, \delta)$ where $1 \leq i$ and $0 \leq \delta \leq d_i$. The configuration corresponding to π is $s_\pi = (q_i, v_i +_{\lambda(q_i)} \delta)$ (with $q_1 = q$ and $v_1 = v$). We denote by $<_\rho$ the strict lexicographical order over positions of ρ.
For basic formulas:

$$\begin{array}{lll}
s \models p & \text{iff} & p \in lab(q) \\
s \models P \bowtie 0 & \text{iff} & v \models P \bowtie 0
\end{array}$$

and inductively:

$$\begin{array}{lll}
s \models \varphi \wedge \psi & \text{iff} & s \models \varphi \text{ and } s \models \psi \\
s \models \neg\varphi & \text{iff} & s \not\models \varphi \\
s \models \mathsf{A}\,\varphi\,\mathsf{U}\,\psi & \text{iff} & \text{for all } \rho \in \mathit{Run}(s),\ \rho \models \varphi\,\mathsf{U}\,\psi \\
s \models \mathsf{E}\,\varphi\,\mathsf{U}\,\psi & \text{iff} & \text{there exists } \rho \in \mathit{Run}(s) \text{ s. t. } \rho \models \varphi\,\mathsf{U}\,\psi \\
\text{with } \rho \models \varphi\,\mathsf{U}\,\psi & \text{iff} & \text{there is a position } \pi \in \rho \text{ s. t. } s_\pi \models \psi \\
 & & \text{and } \forall \pi' <_\rho \pi,\ s_{\pi'} \models \varphi \vee \psi.
\end{array}$$

The automaton $\mathcal{A}$ satisfies ψ (written $\mathcal{A} \models \psi$) if the initial configuration s_0 of $\mathcal{T}_\mathcal{A}$ satisfies ψ. The *model checking* problem asks, given $\mathcal{A}$ and ψ, whether $\mathcal{A} \models \psi$. We now prove the following result.

Theorem 4. *The model checking problem of* TCTL_{int} *over* PolITA *is decidable in time* $(|\mathcal{A}| \cdot |\psi| \cdot d)^{2^{O(n)}}$ *where* n *is the number of clocks in* $\mathcal{A}$ *and* d *the maximal degree of polynomials appearing in* $\mathcal{A}$ *and* ψ.

The automaton built from $\mathcal{A}$ for the reachability problem must be adapted as follows. Given a $\mathsf{TCTL}_{\text{int}}$ formula ψ, we define $\mathit{Poly}(\psi)$ as the set of all polynomials appearing in ψ, *i.e.*, in subformulas of the form $P \bowtie 0$. We now consider $\mathcal{D}_{\mathcal{A},\psi}$, the cylindrical decomposition adapted to $\mathit{Poly}(\mathcal{A}) \cup \mathit{Poly}(\psi)$ and build the associated transition system $\mathcal{R}_{\mathcal{A},\psi}$ as done previously.

Finally, we translate a comparison $P \bowtie 0$ in ψ into a fresh atomic proposition $p_{P\bowtie 0}$. Note that since $\mathcal{D}_{\mathcal{A},\psi}$ is in particular adapted to $\mathit{Poly}(\psi)$, every cell C of $\mathcal{D}_{\mathcal{A},\psi}$ is sign-invariant for P, hence the truth value of $P \bowtie 0$ is constant in C. As a result, we can write $C \models P \bowtie 0$ whenever $v \models P \bowtie 0$ for some $v \in C$, and proposition $p_{P\bowtie 0}$ is true in every state (q, C) where $C \models P \bowtie 0$. Writing $\overline{\psi}$ for the formula obtained from ψ by replacing each $P \bowtie 0$ by $p_{P\bowtie 0}$, we have:

Proposition 4. $\mathcal{A} \models \psi$ *if, and only if,* $\mathcal{R}_{\mathcal{A},\psi} \models \overline{\psi}$.

Since $\overline{\psi}$ is a CTL formula, it can be checked with the usual polynomial time labeling procedure. From the doubly exponential number of cells in the cylindrical decomposition, we obtain the complexity stated in Theorem 4.

6. Expressiveness

We finally focus on the expressiveness of PolITA. After comparing this class with o-minimal hybrid systems and stopwatch automata, we show how to extend it while keeping decidable the above verification problems. For the sake of clarity, in Section 2 we have presented a basic model of PolITA. Here we consider three new features obtained by: (1) including parameters in the expressions of guards and updates, (2) associating with each level a subset of auxiliary clocks, and (3) allowing updates for clocks of levels lower than the current one. Since in the context of ITA, the first two extensions have already been studied in [13] and the third one in [12], our presentation will not be fully formalized.

6.1. Remarks on o-minimal hybrid systems

Following [10], recall that a totally ordered structure $\mathcal{M} = (M, <)$ is o-minimal if every definable subset of M is a finite union of points and open intervals (possibly unbounded). The main property of o-minimal structures is that M^n (for $n \in \mathbb{N} \setminus \{0\}$) admits finite cell decompositions in the same spirit as the particular structure of the real numbers. Observe that [29, 30, 31] provide quantitative bounds on the number of cells for such a decomposition. An o-minimal dynamical system is a pair $(\mathcal{M}, \gamma)$, where $\mathcal{M}$ is o-minimal and $\gamma : M^{k_1} \times M \to M^{k_2}$ (with k_1, k_2 in $\mathbb{N} \setminus \{0\}$) is a definable function of $\mathcal{M}$. The function γ describes the trajectories: M^{k_1} stands for the parameters' space and its second argument represents the time. The target space of γ, M^{k_2}, is the state space.

Finally, an o-minimal hybrid system $\mathcal{H}$ is composed of a finite set of locations, each of which equipped with an o-minimal dynamical system, and such that the transitions between locations are decorated with guards and resets that are subsets in M^{k_2} definable in $\mathcal{M}$. More precisely, in a discrete transition along some edge e from location ℓ_1 to ℓ_2 with guard G_e and reset R_e, a pair (ℓ_1, y_1) with $y_1 \in G_e$ is sent to (ℓ_2, y_2) where y_2 is an arbitrary point in R_e. This strong reset condition implies that finding a finite bisimulation for $\mathcal{H}$ reduces to finding such a bisimulation for each location. These techniques have been used in particular for *Pfaffian hybrid systems* [32], that are special cases of o-minimal hybrid systems where Pfaffian functions like *exp*, trignometrical functions defined in appropriate domains, etc. are added to polynomials. However, in these cases, the algorithm must additionally assume that an oracle deciding the consistency of a Pfaffian systems is given. Note that, up to our knowledge, such an oracle does not exist for the general class of Pfaffian systems.

Moreover, it is shown in [10] that relaxing the assumptions, either on the continuous dynamics or on the discrete transitions, can lead to systems that do not admit finite bisimulations. In particular, this is the case if the transitions are generalized by authorizing some kind of memory for the resets. Therefore, the model of PolITA, like the one of timed automata, are specific subcases of hybrid systems where finite bisimulations exist in spite of the generalized reset conditions, and where no oracle is needed.

6.2. PolITA vs *Timed automata and Stopwatch automata*

By syntax inclusion, PolITA are at least as expressive as ITA. As a direct consequence of the results in [11], there exists a timed language accepted by a PolITA that is not accepted by a TA.

Conversely, there exists a timed language accepted by a timed automaton that is not accepted by any PolITA as defined in Section 2. Consider the language:

$$\begin{aligned} L_1 = \big\{(a,t_1)(b,t_2) \quad \dots \quad & (a,t_{2p+1})(b,t_{2p+2}) \mid p \in \mathbb{N}, \\ & \forall i, 0 \le i \le p,\ t_{2i+1} = i+1 \text{ and } i+1 < t_{2i+2} < i+2, \\ & \forall i, 1 \le i \le p,\ t_{2i+2} - t_{2i+1} < t_{2i} - t_{2i-1}\big\}. \end{aligned}$$

The corresponding untimed language is $(ab)^+$ and in the timed words of L_1, there is an occurrence of a at each time unit and the successive occurrences of b come each time closer to the previous occurrence of a.

It was shown in [12] that no ITA can accept L_1. The proof also holds for PolITA since it is only based on the following hypotheses: (1) there is a single clock per level, (2) at level i, the behavior is only determined by the current state and the values of clocks at levels less or equal than i, and (3) the clock x_i is null at level $j < i$.

Note, however that L_1 is accepted by the extension with auxiliary clocks provided in Section 6.4.

We now show that the class of stopwatch automata (SWA), which also syntactically contains the class of ITA, is incomparable to PolITA. Recall that in a stopwatch automaton, each clock can be active or inactive in every state. Also recall that updates are restricted to resets[2] $x := 0$ and guards are comparisons to a rational constant[3]. In the remainder of the section, we

[2]It is possible to simulate assignments with rational constants, but it does not change expressiveness of the model.

[3]Again, diagonal constraints $x - y \bowtie c$ for $c \in \mathbb{Q}$ can be simulated.

use $+_q$ to denote addition only on stopwatches active in q. We first show the following lemma about runs accepted by a SWA:

Lemma 1. *Let $\rho = (q_0, v_0) \xrightarrow{\delta_0} (q_0, v_0 +_{q_0} \delta_0) \xrightarrow{g_0,a_0,u_0} (q_1, v_1) \cdots$ be a run in a stopwatch automaton. Then there exists $\rho' = (q_0, v_0) \xrightarrow{\delta'_0} (q_0, v_0 +_{q_0} \delta'_0) \xrightarrow{g_0,a_0,u_0} (q_1, v'_1) \cdots$ taking the same discrete transitions as ρ such that $\forall i$, $\delta'_i \in \mathbb{Q}$.*

Proof. We assume that stopwatches are never reset throughout the run. This can be done since one can assume that a reset stopwatch is actually a fresh one. Let X be the set of stopwatches. We consider a system of linear constraints with a variable δ_i per delay and rational coefficients which correspond to all guards appearing in ρ. We define:

$$\gamma_i^x = \begin{cases} 1 \text{ if } x \text{ is active in } q_i \\ 0 \text{ otherwise} \end{cases}$$

Writing $|\rho|$ for the number of discrete transitions in ρ, the system contains the following constraints for $0 \le i < |\rho|$:

$$\delta_i \ge 0 \qquad\qquad \text{and} \qquad\qquad \left(\sum_{\ell=0}^{i} \gamma_\ell^x \cdot \delta_\ell\right)_{x \in X} \models g_i$$

The last constraint may actually denote several constraints if g_i is a conjunctive formula.

Note that since guards have rational coefficients, this system also has rational coefficients. In addition since ρ is an accepted run, this system has a solution $(\delta_0, \dots)$. Also note that for every solution $(\delta'_i)_i$, replacing each delay δ_i with δ'_i in ρ still yields a valid run ρ', since all guards are still respected. It is well-known that when the set of solutions of a system of linear constraints is non empty, there is a solution with rational coefficients, say $(\delta'_i)_i \in \mathbb{Q}^{|\rho|}$. So ρ' is a run with rational delays. $\square$

Proposition 5. *There exists a timed language accepted by a PolITA with a single clock that cannot be accepted by a stopwatch automaton.*

Proof. Consider the PolITA of Figure 5, which accepts the timed language L_2 containing the single word $(a, 1)(b, \sqrt{2})$. Assume that L_2 is accepted by a stopwatch automaton $\mathcal{A}_2$ and let $\rho = (q_0, v_0) \xrightarrow{\delta_0} (q_0, v_0 +_{q_0} \delta_0) \xrightarrow{g_0,a_0,u_0}$

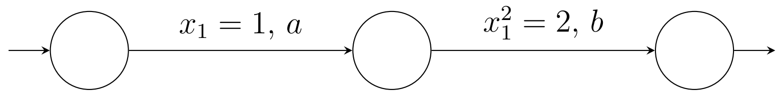


Figure 5: A PolITA whose timed language is not accepted by a stopwatch automaton.

$(q_1, v_1)\cdots$ be a run accepting $(a, 1)(b, \sqrt{2})$. Note that some a_is may actually be ε. Since b occurs at an irrational instant, there is at least an irrational delay before the occurrence of b. By Lemma 1, some run ρ' where all delays are rational is also accepted. Therefore the instant of b in ρ' is rational and cannot be $\sqrt{2}$. □

Furthermore any time rescaling for L_2 does not change this result since either a or b is taken at an irrational instant.

On the other hand, the (untimed) language of a PolITA (and the extensions of Section 6) is regular, as shown by Proposition 3. It is not necessarily the case of (untimed) languages of stopwatch automata [20, 7], hence there are some timed languages accepted by a SWA that are not accepted by any PolITA.

6.3. Parameters

Getting a complete knowledge of a system is often impossible, especially when integrating quantitative constraints. Moreover, even if these constraints are known, when the execution of the system slightly deviates from the expected behavior, due to implementation choices, previously established properties may not hold anymore. Additionally, considering a wide range of values for constants allows for a more flexible and robust design. Introducing parameters instead of concrete values is an elegant way of addressing these three issues. Parametrization however makes verification more difficult. For instance, in timed automata, allowing a single clock to be compared to parameters leads to undecidability of the reachability problem [21].

Suppose that we enlarge PolITA allowing expressions to be polynomials whose set of variables is the union of a set of clocks $\{x_1, \ldots, x_n\}$ and a set of parameters $\{p_1, \ldots, p_k\}$. Then we consider the cylindrical decomposition where the order of variables is $p_1, \ldots, p_k, x_1, \ldots, x_n$. Now assume that the relevant values of parameters are specified by a first-order formula *val*. Then using the cylindrical decomposition, we can answer reachability questions like “for all $p_1 \cdots p_k$ satisfying *val*, is q reachable?” or safety questions like “for all $p_1 \cdots p_k$ satisfying *val*, is q unreachable?”.

6.4. Auxiliary clocks

With each level i, one may associate a set of *auxiliary* clocks Y_i in addition to the *main* clock x_i. Since there are now multiple clocks for some level i, in this PolITA, with every state of level i, is associated an *active* clock in the set $X_i = \{x_i\} \cup Y_i$, specifying which clock evolves with time in this state. Auxiliary clocks may be used in a restrictive setting w.r.t. the main clocks to influence the behavior of the PolITA. Let us detail these restrictions:

- In a guard of a transition outgoing from a state at level i, among auxiliary clocks only those of the level i may occur and they are only be compared between them or with the main clock (i.e. $z \bowtie z'$ with $z, z' \in X_i$);
- In a transition outgoing from state at level i, an auxiliary clock of level i may be updated by another clock of level i (i.e. $y := z$ with $y \in Y_i$ and $z \in X_i$) while the main clock may be updated by an auxiliary clock only if the destination state of the transition is also at level i (i.e. $x_i := y$ with $y \in Y_i$).

The decision procedure works as follows. The cylindrical decomposition does not take into account the auxiliary clocks. However the definition of a class specifies in which interval of level i lies any clock of level i and their relative position for clocks inside the same interval.

Adding auxiliary clocks strictly extends expressiveness of PolITA w.r.t. timed languages. Consider the PolITA of Figure 6 with a single level and single final state q_2. The main clock x is active in all states and y is an auxiliary clock. It is routine to check that the timed language of this automaton is the language L_1 given in section 6.2 above.

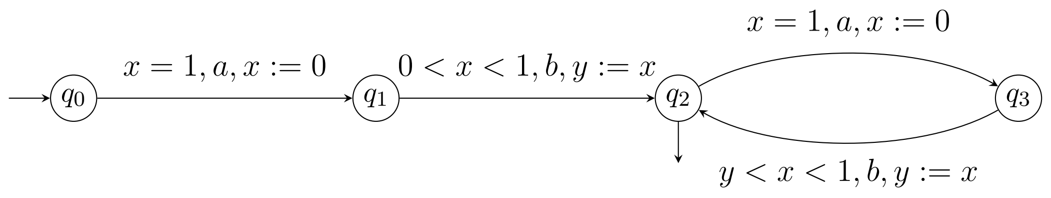


Figure 6: A PolITA with a single level and an auxiliary clock accepting L_1.

6.5. Extended updates

At level i, the value of a clock of level $j < i$ is relevant. So it is interesting to allow updates of such a clock. Again for keeping decidability, such updates have the following restrictions:

- At level i, the main clock of level $j < i$ can only be updated by a polynomial of the main clocks of level less than j: $x_j := P(x_1, \ldots, x_{j-1})$;
- At level i, an auxiliary clock of level $j < i$ may be updated by a clock of level j: $y := z$ with $y \in Y_j$ and $z \in X_j$.

The decision procedure for this extension consists in translating the extended PolITA in a PolITA with the same behavior at level i: (1) delaying the update of clocks of level $j < i$ that should have been done until the current level becomes j and (2) duplicating the states by memorizing the current value of such a clock as an expression of the values of the clock when the level j was left. Furthermore iterating this process, by considering successive updates, only a finite (but exponential) number of duplications is triggered. Guards and updates outgoing from a duplicated state are modified to take into account these expressions.

We illustrate this transformation on the PolITA of Figure 7 that is transformed in the PolITA of Figure 8. The original automaton has only main clocks and the level of the state is indicated inside the state. In the transformed state the superscript '+' means that it corresponds to a state of the original ITA ready to be simulated while the superscript '-' indicates that the delayed updates have to be performed. Let us start with the transition outgoing the state q_0: the update of x_1 is delayed but memorized in the state $[q_2^+, x_1 := 2]$. The transition outgoing from this state corresponds to the transition outgoing from q_2 but in the guard the occurrence of x_1 has been substituted by 2. With this transformation, the update becomes $x_2 := 5$ but since we are at level 3, this update is memorized in state $[q_3^+, x_1 := 2, x_2 := 5]$. The transition from q_3 at level 3 to q_5 at level 2 is split into two transitions in the simulating PolITA. First we enter state $[q_5^-, x_1 := 2, x_2 := 5]$ at level 2 where the active clock is y_2, an auxiliary clock of level 2. Then in null time, due to the guard, we perform the delayed update of x_2, still memorizing the update of x_1 and enter the state $[q_5^+, x_1 := 2]$.

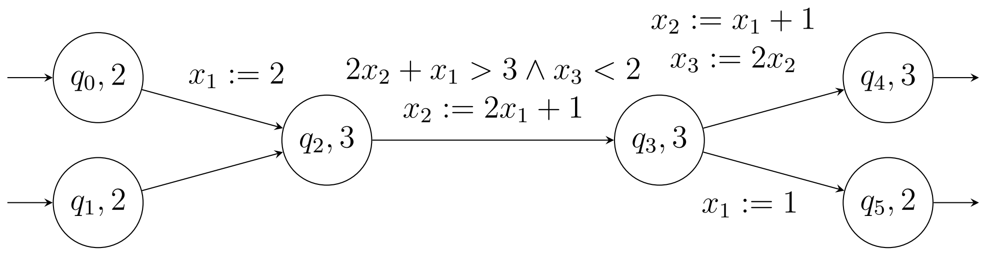


Figure 7: A PolITA containing extended updates of clocks.

7. Conclusion

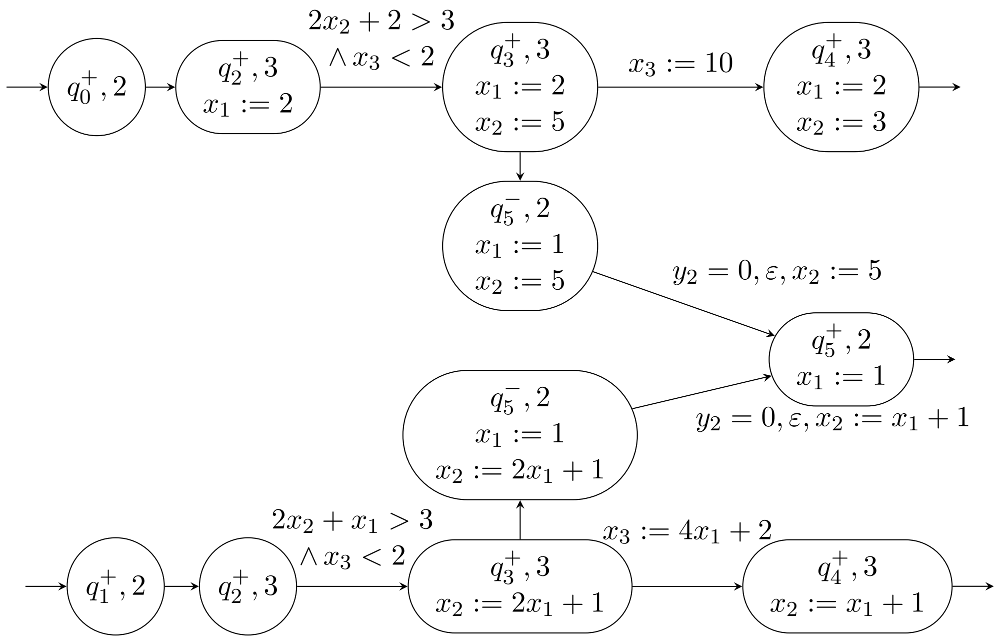
We extend ITA with polynomial expressions on clocks, and prove that reachability and model checking of a timed extension of CTL are decidable using the cylindrical algebraic decomposition. We also show that an on-the-fly construction of a class automaton is possible during the lifting phase of this decomposition. We discuss extensions of PolITA that increase expressiveness while preserving the decidability results, and show how the class PolITA relates to other hybrid models.

An implementation[4] (in Python, using the Sage library) was produced as a proof-of-concept, showing the practical feasibility of the reachability algorithm. A further step on the applicative side could be to look for biological systems where PolITA would be an adequate model.

Since the construction still suffers from the doubly exponential complexity of the cylindrical decomposition, we plan to investigate if recent methods proposed in [33] with a lower complexity could be used to achieve reachability, possibly for a restricted version of PolITA. Another direction would be to enlarge the class of functions (like those studied in [34]) labeling guards and updates, still ensuring decidability of the reachability problem.

Acknowledgements. This work was partially supported by LIP6 project Parasol. We would like to thank all anonymous reviewers for their insightful comments and suggestions. We would also like to thank Rémi Garnier and Mathieu Huot for implementing the reachability algorithm.

[4]Available at `https://github.com/MathieuHuot/D-composition-Cylindrique`.



$q_0^+, 2$
$q_2^+, 3$ $x_1 := 2$
$2x_2 + 2 > 3$ $\wedge x_3 < 2$
$q_3^+, 3$ $x_1 := 2$ $x_2 := 5$
$x_3 := 10$
$q_4^+, 3$ $x_1 := 2$ $x_2 := 3$
$q_5^-, 2$ $x_1 := 1$ $x_2 := 5$
$y_2 = 0, \varepsilon, x_2 := 5$
$q_5^+, 2$ $x_1 := 1$
$q_5^-, 2$ $x_1 := 1$ $x_2 := 2x_1 + 1$
$y_2 = 0, \varepsilon, x_2 := x_1 + 1$
$2x_2 + x_1 > 3$ $\wedge x_3 < 2$
$q_1^+, 2$
$q_2^+, 3$
$q_3^+, 3$ $x_2 := 2x_1 + 1$
$x_3 := 4x_1 + 2$
$q_4^+, 3$ $x_2 := x_1 + 1$

Figure 8: A PolITA equivalent to the PolITA of Figure 7.

References

[1] B. Bérard, S. Haddad, C. Picaronny, M. Safey El Din, M. Sassolas, Polynomial Interrupt Timed Automata, in: Proceedings of the 9th International Workshop on Reachability problems (RP'15), Vol. 9328 of LNCS, Springer, 2015, pp. 20–32.

[2] R. Grossman, A. Nerode, A. Ravn, H. Rischel (Eds.), Hybrid systems, Vol. 736 of LNCS, Springer, 1993.

[3] A. Tiwari, G. Khanna, Series of abstractions for hybrid automata, in: C. J. Tomlin, M. R. Greenstreet (Eds.), Hybrid Systems: Computation and Control HSCC, Vol. 2289 of LNCS, Springer, 2002, pp. 465–478.

[4] R. Alur, D. L. Dill, A theory of timed automata, Theoretical Computer Science 126 (1994) 183–235.

[5] T. A. Henzinger, P. W. Kopke, A. Puri, P. Varaiya, What's decidable about hybrid automata?, Journal of Computer and System Science 57 (1) (1998) 94–124.

[6] E. Asarin, O. Maler, A. Pnueli, Reachability analysis of dynamical systems having piecewise-constant derivatives, Theoretical Computer Science 138 (1) (1995) 35–65.

[7] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, S. Yovine, The algorithmic analysis of hybrid systems, Theoretical Computer Science 138 (1995) 3–34.

[8] G. Lafferriere, G. J. Pappas, S. Sastry, O-minimal hybrid systems, Mathematics of Control, Signals, and Systems 13 (1) (2000) 1–21.

[9] R. Alur, T. A. Henzinger, G. Lafferriere, G. J. Pappas, Discrete abstractions of hybrid systems, Proceedings of the IEEE 88 (7) (2000) 971–984.

[10] T. Brihaye, C. Michaux, On the expressiveness and decidability of o-minimal hybrid systems, Journal of Complexity 21 (4) (2005) 447–478.

[11] B. Bérard, S. Haddad, Interrupt timed automata, in: Proceedings of the 12th International Conference on Foundations of Software Science and Computational Structures (FoSSaCS'09), Vol. 5504 of LNCS, Springer, York, UK, 2009, pp. 197–211.

[12] B. Bérard, S. Haddad, M. Sassolas, Interrupt timed automata: Verification and expressiveness, Formal Methods in System Design 40 (1) (2012) 41–87.

[13] B. Bérard, S. Haddad, A. Jovanovič, D. Lime, Parametric interrupt timed automata, in: Proceedings of the 7th Workshop on Reachability Problems (RP'13), Vol. 8169 of LNCS, Springer, 2013, pp. 59–69.

[14] M. Müller-Olm, H. Seidl, Computing polynomial program invariants, Information Processing Letters 91 (5) (2004) 233–244.

[15] D. Babic, B. Cook, A. J. Hu, Z. Rakamaric, Proving termination of nonlinear command sequences, Formal Asp. Comput. 25 (3) (2013) 389–403.

[16] A. Finkel, S. Göller, C. Haase, Reachability in register machines with polynomial updates, in: K. Chatterjee, J. Sgall (Eds.), Proc. of 38th Int. Symp. on Mathematical Foundations of Computer Science 2013 - 38th International Symposium, MFCS'13, Klosterneuburg, Austria, August

26-30., Vol. 8087 of Lecture Notes in Computer Science, Springer, 2013, pp. 409–420.

[17] C. Piazza, M. Antoniotti, V. Mysore, A. Policriti, F. Winkler, B. Mishra, Algorithmic algebraic model checking I: Challenges from systems biology, in: K. Etessami, S. K. Rajamani (Eds.), Proceedings of the 17th International Conference on Computer Aided Verification (CAV'05), Vol. 3576 of LNCS, Springer, 2005, pp. 5–19.

[18] M. Adélaïde, O. F. Roux, Using cylindrical algebraic decomposition for the analysis of slope parametric hybrid automata, in: Proceedings of the 6th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'00), Vol. 1926 of LNCS, Springer, 2000, pp. 252–263.

[19] R. Alur, C. Courcoubetis, D. L. Dill, Model-checking in dense real-time, Information and Computation 104 (1) (1993) 2–34.

[20] F. Cassez, K. G. Larsen, The impressive power of stopwatches, in: Proceedings of the 11th International Conference on Concurrency Theory (CONCUR'00), Vol. 1877 of LNCS, Springer, 2000, pp. 138–152.

[21] J. S. Miller, Decidability and complexity results for timed automata and semi-linear hybrid automata, in: Proceedings of the 3rd International Workshop on Hybrid Systems: Computation and Control (HSCC'00), Vol. 1790 of LNCS, Springer, 2000, pp. 296–309.

[22] G. E. Collins, Quantifier elimination for real closed fields by cylindrical algebraic decomposition, in: Automata Theory and Formal Languages 2nd GI Conference, Vol. 33 of LNCS, Springer Berlin Heidelberg, 1975, pp. 134–183.

[23] S. Basu, R. Pollack, M.-F. Roy, Algorithms in Real Algebraic Geometry, Springer, 2006.

[24] B. Bérard, S. Haddad, C. Picaronny, M. Safey El Din, M. Sassolas, Polynomial interrupt timed automata, CoRR abs/1504.04541 (2015) 1–53.

[25] M. Ben-Or, D. Kozen, J. Reif, The complexity of elementary algebra and geometry, in: Proceedings of the 16th Annual ACM Symposium on Theory of Computing (STOC'84), ACM, 1984, pp. 457–464.

[26] E. A. Emerson, J. Y. Halpern, Decision procedures and expressiveness in the temporal logic of branching time, in: Proceedings of the 14th annual ACM Symposium on the Theory of Computing (STOC'82), ACM, 1982, pp. 169–180.

[27] J.-P. Queille, J. Sifakis, Specification and verification of concurrent systems in CESAR, in: Proceedings of the 5th International Symposium on Programming, Vol. 137 of LNCS, Springer, London, UK, 1982, pp. 337–351.

[28] T. A. Henzinger, X. Nicollin, J. Sifakis, S. Yovine, Symbolic model checking for real-time systems, Information and Computation 111 (2) (1994) 193–244.

[29] A. Gabrielov, N. Vorobjov, Complexity of stratification of semi-Pfaffian sets, Discrete & Computational Geometry 14 (1) (1995) 71–91.

[30] A. Gabrielov, N. Vorobjov, Complexity of computations with Pfaffian and Noetherian functions, in: Proc. of Normal Forms, Bifurcations and Finiteness Problems in Differential Equations, Vol. 137 of NATO Science, Kluwer, 2004, pp. 211–250.

[31] A. Gabrielov, N. Vorobjov, T. Zell, Betti numbers of semialgebraic and sub-Pfaffian sets, Journal of the London Mathematical Society 69 (01) (2004) 27–43.

[32] M. Korovina, N. Vorobjov, Upper and lower bounds on sizes of finite bisimulations of Pfaffian hybrid systems, in: Logical Approaches to Computational Barriers, Springer, 2006, pp. 267–276.

[33] H. Hong, M. Safey El Din, Variant quantifier elimination, Journal of Symbolic Computation 47 (7) (2012) 883–901.

[34] D. J. Miller, Constructing o-minimal structures with decidable theories using generic families of functions from quasianalytic classes, CoRR abs/1008.2575 (2010) 1–42.