Parametric Interrupt Timed Automata[⋆]

B. Bérard[1], S. Haddad[2], A. Jovanović[3], and D. Lime[3]

[1] Université Pierre & Marie Curie, LIP6/MoVe, CNRS, Paris, France
[2] ENS Cachan, LSV, CNRS, INRIA, Cachan, France
[3] LUNAM Université, École Centrale de Nantes, IRCCyN, CNRS, Nantes, France

Abstract. Parametric reasoning is particularly relevant for timed models, but very often leads to undecidability of reachability problems. We propose a parametrised version of Interrupt Timed Automata (an expressive model incomparable to Timed Automata), where polynomials of parameters can occur in guards and updates. We prove that different reachability problems, including robust reachability, are decidable for this model, and we give complexity upper bounds for a fixed or variable number of clocks and parameters.

1 Introduction

Parametric verification. 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 behaviour, 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. Parametrisation however makes verification more difficult. Besides, it raises new problems like parameter synthesis, *i.e.* finding the set (or a subset) of values for which some property holds.
Parameters for timed models. Among quantitative features, parametric reasoning is particularly relevant for timing requirements, like network delays, time-outs, response times or clock drifts.

Pioneering work on parametric real time reasoning was presented in [1] for the now classical model of timed automata [2] with parameter expressions replacing the constants to be compared with clock values. Since then, many studies have been devoted to the parametric verification of timed models [3,4,5], mostly establishing undecidability results for questions like parametric reachability, even for a small number of clocks or parameters.

[⋆] This work has been supported by project ImpRo ANR-2010-BLAN-0317

Relaxing completeness requirement or guaranteed termination, several methods and tools have been developed for parameter synthesis in timed automata [6,7,8], as well as in hybrid automata [9,10]. Another research direction consists in defining subclasses of parametric timed models for which some problems become decidable [11,12,13]. Unfortunately, these subclasses are severely restricted. It is then a challenging issue to define expressive parametric timed models where reachability problems are decidable.
Contributions. The model of interrupt timed automata (ITA) [14,15] was proposed as a subclass of hybrid automata, incomparable with the class of timed automata, where task interruptions are taken into account. Hence ITA are particularly suited for the modelling of scheduling with preemption.

We propose a parametric version of ITA where polynomial parameter expressions can be combined with clock values both as additive and multiplicative coefficients. The multiplicative setting is much more expressive and useful in practice, for instance to model clock drifts. We prove that reachability in parametric ITA is decidable as well as its robust variant, an important property for implementation issues. To the best of our knowledge, this is the first time such a result has been obtained for a model including a multiplicative parametrisation. Furthermore, we establish upper bounds for the algorithms complexity: 2EXSPACE and PSPACE when the number of clocks is fixed, which become respectively 2EXPTIME and PTIME for additive parametrisation, when the number of clocks and parameters is fixed. Our technique combines the construction of symbolic class automata from the ITA case and the first order theory of real numbers. Finally, considering only additive parametrisation, we reduce reachability to the same problem in basic ITA.
Outline. The parametric ITA model is introduced in Section 2 and decision procedures are presented in Section 3 with complexity analysis. We conclude and give some perpectives for this work in Section 4. All proofs are given in the appendix.

2 Parametric Interrupt Timed Automata

2.1 Notations

The sets of natural, rational and real numbers are denoted respectively by $\mathbb{N}$, $\mathbb{Q}$ and $\mathbb{R}$. Given two sets A, B, we denote by $\mathcal{P}ol(A, B)$, the set of polynomials with variables in A and coefficients in B. We also denote by $\mathcal{L}in(A, B)$ the subset of polynomials with degree at most one and by

$\mathcal{F}rac(A, B)$, the set of rational functions with variables in A and coefficients in B (i.e. quotients of polynomials).

Clock and parameter constraints. Let X be a finite set of clocks and let P be a finite set of parameters. An expression over clocks is an element $\sum_{x \in X} a_x \cdot x + b$ of $\mathcal{L}in(X, \mathcal{P}ol(P, \mathbb{Q}))$. In the sequel we also consider two other sets of expressions: $\mathcal{L}in(X, \mathbb{Q})$ and $\mathcal{L}in(X \cup P, \mathbb{Q})$. The former is the subset of expressions without parameters while the latter can be seen as a subset of expressions where $a_x \in \mathbb{Q}$ for all $x \in X$ and $b \in \mathcal{L}in(P, \mathbb{Q})$. We denote by $\mathcal{C}(X, P)$ the set of constraints obtained by conjunctions of atomic propositions of the form $C \bowtie 0$, where C is an expression of $\mathcal{L}in(X, \mathcal{P}ol(P, \mathbb{Q}))$ and $\bowtie \in \{>, \geq, =, \leq, <\}$.

Updates and valuations. An *update* is a conjunction (over X) of assignments of the form $x := C_x$, where x is a clock and $C_x \in \mathcal{L}in(X, \mathcal{P}ol(P, \mathbb{Q}))$. The set of updates is written $\mathcal{U}(X, P)$. For an expression C and an update u, the expression $C[u]$ is obtained by "applying" u to C, *i.e.*, substituting each x by C_x in C, if $x := C_x$ is the update for x in u. Observe that an update is performed simultaneously on all clocks. For instance, for clocks $X = \{x_1, x_2\}$, parameters $P = \{p_1, p_2, p_3\}$, expression $C = p_2x_2 - 2x_1 + 3p_1$ and the update u defined by $x_1 := 1 \wedge x_2 := p_3x_1 + p_2$, applying u to C yields the expression $C[u] = p_2p_3x_1 + p_2^2 + 3p_1 - 2$.

A *clock valuation* is a mapping $v : X \mapsto \mathcal{P}ol(P, \mathbb{R})$, with $\mathbf{0}$ the valuation where all clocks have value 0. For a valuation v and an expression $C \in \mathcal{L}in(X, \mathcal{P}ol(P, \mathbb{Q}))$, $v(C) \in \mathcal{P}ol(P, \mathbb{R})$ is obtained by evaluating C w.r.t. v. Given an update u and a valuation v, the valuation $v[u]$ is defined by $v[u](x) = v(C_x)$ for x in X if $x := C_x$ is the update for x in u. For instance, let $X = \{x_1, x_2, x_3\}$ be a set of three clocks. For valuation $v = (2p_2, 1.5, 3p_1^2)$ and update u defined by $x_1 := 1 \wedge x_2 := x_2 \wedge x_3 := p_1x_3 - x_1$, applying u to v yields the valuation $v[u] = (1, 1.5, 3p_1^3 - 2p_2)$.

A parameter valuation is a mapping $\pi : P \mapsto \mathbb{R}$. For a parameter valuation π and an expression $C \in \mathcal{L}in(X, \mathcal{P}ol(P, \mathbb{Q}))$, $\pi(C) \in \mathcal{L}in(X, \mathbb{R})$ is obtained by evaluating C w.r.t. π. If $C \in \mathcal{P}ol(P, \mathbb{Q})$, then $\pi(C) \in \mathbb{R}$. Given a parameter valuation π, a clock valuation v and an expression $C \in \mathcal{L}in(X, \mathcal{P}ol(P, \mathbb{Q}))$ we write $\pi, v \models C \bowtie 0$ when $\pi(v(C)) \bowtie 0$.

2.2 Parametric Interrupt Timed Automata

Definitions. The behaviour of an ITA can be viewed as the one of an operating system with interrupt levels. At a given level, exactly one clock is active (rate 1), while the clocks at lower levels are suspended (rate 0), and the clocks at higher levels are not yet activated and thus contain value 0. The enabling conditions of transitions, called *guards*, are constraints in $\mathcal{L}in(X, \mathbb{Q})$ over clocks of levels lower than or equal to the current level. Transitions can *update* the clock values. If the transition decreases (resp. increases) the level, then each clock which is relevant after (resp. before) the transition can either be left unchanged or take a linear expression of clocks of strictly lower level.

Parametric ITA include parameters in guards and updates.

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

- *Σ is a finite alphabet, P is a finite set of parameters,*
- *Q is a finite set of states, q_0 is the initial state,*
- *$X = \{x_1, \ldots, x_n\}$ consists of n interrupt clocks,*
- *the mapping $\lambda : Q \to \{1, \ldots, n\}$ associates with each state its level; we assume $\lambda(q_0) = 1$, $X_{\lambda(q)} = \{x_i \mid i \leq \lambda(q)\}$ is the set of* relevant *clocks at this level and $x_{\lambda(q)}$ is called the* active *clock in state q;*
- *$\Delta \subseteq Q \times \mathcal{C}(X, P) \times (\Sigma \cup \{\varepsilon\}) \times \mathcal{U}(X, P) \times Q$ 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 constraint in $\mathcal{C}(X_k, P)$ (using only 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 is either of the form $\sum_{j=1}^{i-1} a_j x_j + b$ or $C_i = x_i$ (unchanged clock value) and for $i > k'$, $C_i = 0$;*
  - *if $k \leq k'$ then for $1 \leq i \leq k$, C_i is of the form $\sum_{j=1}^{i-1} a_j x_j + b$ or $C_i = x_i$, and for $i > k$, $C_i = 0$.*

An ITA is a PITA with $P = \emptyset$. When all expressions occurring in guards and updates are in $\mathcal{L}in(X \cup P, \mathbb{Q})$, the PITA is said to be *additively parametrised*, in contrast to the general case, which is called *multiplicatively parametrised.*

We give a transition system describing the semantics of a PITA w.r.t. a parameter valuation π. A configuration (q, v) consists of a state q of the PITA and a clock valuation v.

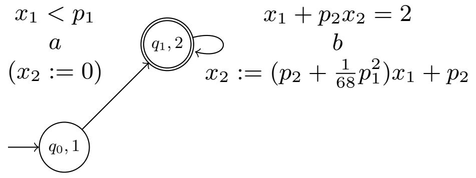
Definition 2. *The semantics of a PITA $\mathcal{A}$ w.r.t. a parameter valuation π is defined by the (timed) transition system $\mathcal{T}_{\mathcal{A},\pi} = (S, s_0, \to)$. The set of configurations is $S = \{(q, v) \mid q \in Q,\ v \in \mathbb{R}^X\}$, with initial configuration $s_0 = (q_0, \mathbf{0})$. The relation $\to$ on S consists of two types of steps:*

Time steps: *Only the active clock in a state can evolve, all other clocks are suspended. For a state q with active clock $x_{\lambda(q)}$, a time step of* *duration d is defined by $(q,v) \xrightarrow{d} (q,v')$ with $v'(x_{\lambda(q)}) = v(x_{\lambda(q)}) + d$ and $v'(x) = v(x)$ for any other clock x. We write $v' = v +_q d$.*

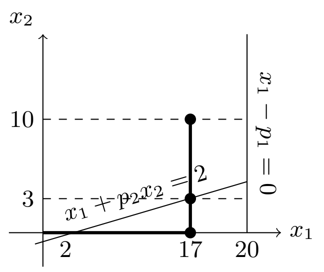
Discrete steps: *A discrete step $(q,v) \xrightarrow{e} (q',v')$ can occur for some transition $e = q \xrightarrow{\varphi,a,u} q'$ in Δ such that $\pi, v \models \varphi$ and $v'(x) = \pi(v[u](x))$.*

A *run* of $\mathcal{A}$ is a finite path in the transition system $\mathcal{T}_{\mathcal{A},\pi}$, for some parameter valuation π, where (possibly null) time steps and discrete steps alternate. A state $q \in Q$ is *reachable* from q_0 for π if there is a path from $(q_0, \mathbf{0})$ to (q,v) in $\mathcal{T}_{\mathcal{A},\pi}$, for some valuation v.

Example 1. A PITA $\mathcal{A}_1$ is depicted in Fig. 1(a), with two interrupt levels. A possible trajectory, symbolically presented as: $(q_0,0,0) \xrightarrow{17} (q_0,17,0) \xrightarrow{a} (q_1,17,0) \xrightarrow{3} (q_1,17,3) \xrightarrow{b} (q_1,17,18p_2 + \frac{17}{68}p_1^2)$, is to stay in state q_0 until the value of x_1 increases from 0 to 17, giving the constraint $p_1 > 17$ for firing a. After transition a occurs, the value of x_1 is frozen in state q_1, while x_2 increases. When x_2 reaches 3, the only value for p_2 satisfying the guard $x_1 + p_2x_2 = 2$ is $p_2 = -5$. After the occurrence of transition b, x_2 is updated to $x_2 = 18p_2 + \frac{17}{68}p_1^2$. Let us fix the parameter valuation π: $p_1 = 20$ and $p_2 = -5$. Hence, the clock x_2 is updated to $x_2 = 10$. A geometric view of this trajectory w.r.t. π is given in Fig. 1(b).



(a) A PITA $\mathcal{A}_1$ with two interrupt levels



(b) A possible trajectory in $\mathcal{A}_1$

Fig. 1. An example of PITA and a possible execution

Problems. We consider here reachability problems for PITA. Let $\mathcal{A}$ be a PITA with initial state q_0 and q be a state of $\mathcal{A}$. The *Existential (resp. Universal) Reachability Problem* asks whether q is reachable from q_0 for some (resp. all) parameter valuation(s). *Scoped* variants of these problems are obtained by adding as input a set of parameter valuations given by either a first order formula of the reals or a polyhedral constraint. The *Robust Reachability Problem* asks whether there exists a parameter valuation π and a real $\varepsilon > 0$ such that for all π' with $\|\pi - \pi'\|_\infty < \varepsilon$, q is reachable from q_0 for π'. When satisfied, this last property ensures that small parameter perturbations do not modify the reachability result.

3 Reachability Analysis

To decide reachability for a PITA $\mathcal{A}$, we build a finite family of class automata related to a finite partition of parameter valuations. Any item of this partition is specified by a satisfiable first-order formula over $(\mathbb{R}, +, \times)$ with the parameters as variables. The specification of classes depends on the expressions occurring in the guards and updates of $\mathcal{A}$. In any of them, a class is defined by a state q and a family $\{\preceq_k\}_{k \leq \lambda(q)}$ of preorders, where $\preceq_k$ orders E_k, a set of expressions in $\mathcal{L}in(X, \mathcal{F}rac(P, \mathbb{Q}))$. The formulas defining the partition are based on E_1 and a set of polynomials $PolPar$.

3.1 Construction of $PolPar$ and $\{E_k\}_{k \leq n}$

The *normalisation* is an operation relative to some level k, operating on an expression. The resulting expressions are obtained by first forgetting terms related to clocks from levels above and then operating transformations depending on the leading coefficient of the truncated expression.

Definition 3 (Normalisation). *Let $k \leq n$ be some level and let $C = \sum_{i \leq n} a_i x_i + b$ be an expression in $\mathcal{L}in(X, \mathcal{F}rac(P, \mathbb{Q}))$, with $a_k = \frac{r_k}{s_k}$, for some r_k and s_k in $\mathcal{P}ol(P, \mathbb{Q})$. The k-*normalisation *of C produces the following expressions:*

– $\texttt{lead}(C, k) = r_k$*;*
– *if* $\texttt{lead}(C, k) \notin \mathbb{Q} \setminus \{0\}$*,* $\texttt{comp}(C, k) = \sum_{i<k} a_i x_i + b$*;*
– *If* $\texttt{lead}(C, k) \neq 0$ *then* $\texttt{compnorm}(C, k) = -\sum_{i<k} \frac{a_i}{a_k} x_i - \frac{b}{a_k}$.

The first expression is the numerator of the coefficient of x_k, the active clock at level k, while the other two expressions could need to be compared to the value of x_k or 0 when some transition is fired at level k, assuming that $C \bowtie 0$ occurs in its guard. If $\texttt{lead}(C, k)$ is equal to 0 (resp. is a non constant polynomial), then $\texttt{comp}(C, k)$ must (resp. could) be compared to 0 (resp. when this polynomial has value 0). Otherwise, x_k must or could be compared to $\texttt{compnorm}(C, k)$.

The construction of $PolPar$ and $\{E_k\}_{k \leq n}$ proceeds top down from level n to level 1 after initialising $PolPar = \emptyset$ and $E_k = \{x_k, 0\}$ for all k. When handling level k, we add new terms to E_i for $1 \leq i \leq k$.

1. At level k the first step consists in adding new expressions to E_k and new polynomials to $PolPar$. More precisely, let C be any expression occurring in a guard of an edge leaving a state of level k. We add $\texttt{lead}(C, k)$ to $PolPar$ when it does not belong to $\mathbb{Q}$ and we add $\texttt{comp}(C, k)$ and $\texttt{compnorm}(C, k)$ to E_k when they are defined.

2. The second step consists in iterating the following procedure until no new term is added to any E_i for $1 \leq i \leq k$.
   (a) Let $q \xrightarrow{\varphi,a,u} q'$ with $\lambda(q) \geq k$ and $\lambda(q') \geq k$, and let $C \in E_k$. Then we add $C[u]$ to E_k (recall that $C[u]$ is the expression obtained by applying update u to C).
   (b) Let $q \xrightarrow{\varphi,a,u} q'$ with $\lambda(q) < k$ and $\lambda(q') \geq k$. Let C and C' be two different expressions in E_k. We compute $C'' = C[u] - C'[u]$, choosing an arbitrary order between C and C' in order to avoid redundancy. Then we proceed with C'' w.r.t. $\lambda(q)$ as done for C w.r.t. k in step 1 above.

Example 2. For the automaton of Fig. 1(a), initially, we have $PolPar = \emptyset$, $E_1 = \{x_1, 0\}$ and $E_2 = \{x_2, 0\}$. We start with level $k = 2$. In step 1, we consider the single edge leaving the state of level 2. The expression occurring in its guard is $C_2 = p_2x_2+x_1-2$. We compute $\texttt{lead}(C_2, 2) = p_2$, $\texttt{comp}(C_2, 2) = x_1-2$, and $\texttt{compnorm}(C_2, 2) = -\frac{x_1-2}{p_2}$. We obtain $PolPar = \{p_2\}$ and $E_2 = \{x_2, 0, x_1 - 2, -\frac{x_1-2}{p_2}\}$.

We proceed with step 2(a) and consider again the same edge. We apply its update $x_2 := (p_2 + \frac{1}{68}p_1^2)x_1 + p_2$ to every expression of E_2 that contains x_2, add them to E_2, and thus obtain $E_2 = \{x_2, 0, x_1 - 2, -\frac{x_1-2}{p_2}, (p_2 + \frac{1}{68}p_1^2)x_1 + p_2\}$.

Step 2(b) considers the single edge from q_0 to q_1 increasing from a level lower than 2 to a level greater than or equal to 2. We compute the differences between any two expressions from E_2 (after applying an update of the edge) and normalise them. Similarly to step 1, we update $PolPar$ and E_1, which yields: $PolPar = \{p_2, p_2 + 1, 1 - p_2 - \frac{1}{68}p_1^2, -p_2^2 - \frac{1}{68}p_1^2p_2 - 1\}$ and $E_1 = \{x_1, 0, 2, -\frac{2(p_2+1)}{p_2}, -2 - p_2, \frac{2+p_2}{1-p_2-\frac{1}{68}p_1^2}, \frac{2-p_2^2}{p_2}, \frac{2-p_2^2}{1+p_2^2+\frac{1}{68}p_1^2p_2}\}$.

We continue with level 1. Since the guard of the considered edge is $x_1 - p_1 < 0$, there is no term to add to $PolPar$.
We add $\texttt{compnorm}(C_1, 1) = p_1$ to E_1. As a result, we obtain:
$E_1 = \{x_1, 0, 2, -\frac{2(p_2+1)}{p_2}, -2 - p_2, \frac{2+p_2}{1-p_2-\frac{1}{68}p_1^2}, \frac{2-p_2^2}{p_2}, \frac{2-p_2^2}{1+p_2^2+\frac{1}{68}p_1^2p_2}, p_1\}$,
$E_2 = \{x_2, 0, x_1 - 2, -\frac{x_1-2}{p_2}, (p_2 + \frac{1}{68}p_1^2)x_1 + p_2\}$ and,
$PolPar = \{p_2, p_2 + 1, 1 - p_2 - \frac{1}{68}p_1^2, -p_2^2 - \frac{1}{68}p_1^2p_2 - 1\}$.

Lemma 1 below is used for the class automata construction. Its proof is obtained by a straightforward examination of the above procedure. The other two lemmata are related to the termination and complexity of this procedure and used in the computation of the upper bound of the reachability algorithm. In lemma 3, we give bounds for the size of integers and polynomials produced by the previous construction, since our algorithms manipulate rationals (resp. rational functions) as pairs of integers (resp. polynomials).

Lemma 1. *Let C belong to E_k for some k and $c = \frac{r}{s}$ be a coefficient of C with $s \notin \mathbb{Q}$. Then s is the product of items of $PolPar$ up to some constant in $\mathbb{Q} \setminus \{0\}$.*

Lemma 2. *The construction procedure of $\{E_k\}_{k \leq n}$ terminates and the size of every E_k is bounded by $(2E+2)^{2^{n(n-k+1)}+1}$ where E is the number of atomic propositions in edges of the PITA.*

Lemma 3. *Let b (resp. d) be the number of bits of an integer constant (resp. the degree of a polynomial) occurring in an expression of $PolPar$ or some E_k. Then $b \leq (n+2)!2^n b_0$ (resp. $d \leq (n+2)!d_0$), where b_0 (resp. d_0) is the maximal number of bits for integers (resp. degree of polynomials) occurring in the PITA.*

3.2 Construction of class automata for PITA

Class definition. Let $\mathcal{A}$ be a PITA. Starting from the (finite) set $PolPar$, we first consider the partition obtained by splitting the set of parameter valuations according to the positions of the polynomials in $PolPar$ with respect to 0. Thus, if $PolPar$ contains p elements, there are at most 3^p parameter regions in the partition. Then, we compute the subset of non empty regions (by solving an existential formula of the first-order theory of reals).

Given a non empty parameter region *preg*, we consider the following subset of E_k for $1 \leq k \leq n$: $E_{k,preg} = \{C \in E_k \mid \text{the denominators of coefficients of } C \text{ are non null in } preg\}$. Due to Lemma 1, these subsets are obtained by examining the specification of *preg*.

Observe that expressions in $E_{1,preg} \setminus \{x_1\}$ belong to $\mathcal{F}rac(P, \mathbb{Q})$. We now refine *preg* according to a linear pre-order $\preceq_1$ on the set of expressions $E_{1,preg} \setminus \{x_1\}$ which is satisfiable within *preg*. We denote this refined region by $(preg, \preceq_1)$ and we build a class automaton $\mathcal{R}(preg, \preceq_1)$ for any such pair.

In $\mathcal{R}(preg, \preceq_1)$, a state, called a *class*, is a syntactical representation of a subset of reachable configurations. More precisely, a class is defined as a pair $R = (q, \{\preceq_k\}_{1 \leq k \leq \lambda(q)})$ where q is a state of $\mathcal{A}$ and $\preceq_k$ is a total preorder over $E_{k,preg}$, for $1 \leq k \leq \lambda(q)$. The class R describes a subset of configurations in $\mathcal{T}_{\mathcal{A},\pi}$, for a parameter valuation $\pi \in (preg, \preceq_1)$:

$$[\![R]\!]_\pi = \{(q,v) \mid \forall k \leq \lambda(q)\ \forall (g,h) \in E_{k,preg},\ \pi(v(g)) \leq \pi(v(h)) \text{ iff } g \preceq_k h\}$$

The initial state of this automaton is defined by the class R_0, such that $[\![R_0]\!]_\pi$ contains $(q_0, \mathbf{0})$, which can be straightforwardly determined by extending $\preceq_1$ to $E_{1,preg}$ with $x_1 = 0$.

As usual, there are two kinds of transitions in $\mathcal{R}(preg, \preceq_1)$, corresponding to discrete steps and time steps.

Discrete step. Let $R = (q, \{\preceq_i\}_{1 \leq i \leq \lambda(q)})$ and $R' = (q', \{\preceq'_i\}_{1 \leq i \leq \lambda(q')})$ be two classes. There is a transition $R \xrightarrow{e} R'$ for a transition $e : q \xrightarrow{\varphi,a,u} q'$ if for some $\pi \in (preg, \preceq_1)$, there is some $(q, v) \in [\![R]\!]_\pi$ and $(q', v') \in [\![R']\!]_\pi$ such that $(q, v) \xrightarrow{e} (q', v')$. In this case, for all $(q, v) \in [\![R]\!]_\pi$ there is a $(q', v') \in [\![R']\!]_\pi$ such that $(q, v) \xrightarrow{e} (q', v')$. We prove in the sequel that the existence of transition $R \xrightarrow{e} R'$ is independent of $\pi \in (preg, \preceq_1)$ and of $(q, v) \in [\![R]\!]_\pi$. It can be decided as follows.

Firability condition. Write $\varphi = \bigwedge_{j \in J} C_j \bowtie_j 0$. For a given j, let us write $C_j = \sum_{i \leq \lambda(q)} a_i x_i + b$. We consider three cases.
• **Case** $a_{\lambda(q)} = 0$**.** Then $C_j = \texttt{comp}(C_j, \lambda(q)) \in E_{\lambda(q),preg}$ and using the positions of 0 and C_j w.r.t. $\preceq_{\lambda(q)}$, we can decide whether $C_j \bowtie_j 0$.
• **Case** $a_{\lambda(q)} \in \mathbb{Q} \backslash \{0\}$**.** Then $\texttt{compnorm}(C_j, \lambda(q)) \in E_{\lambda(q),preg}$, hence using the sign of $a_{\lambda(q)}$ and the positions of $x_{\lambda(q)}$ and $\texttt{compnorm}(C_j, \lambda(q))$ w.r.t. $\preceq_{\lambda(q)}$, we can decide whether $C_j \bowtie_j 0$.
• **Case** $a_{\lambda(q)} \notin \mathbb{Q}$**.** According to the specification of *preg*, we know the sign of $a_{\lambda(q)}$ as it belongs to *PolPar*. In case $a_{\lambda(q)} = 0$, we decide as in the first case. Otherwise, we decide as in the second case.

Successor definition. R' is defined as follows.
Let $k \leq \lambda(q')$ and $g', h' \in E_{k,preg}$.

1. Either $k \leq \lambda(q)$, by step 2(a) of the construction, $g'[u], h'[u] \in E_{k,preg}$. Then $g' \preceq'_k h'$ iff $g'[u] \preceq_k h'[u]$.
2. Or $k > \lambda(q)$, let $D = g'[u] - h'[u] = \sum_{i \leq \lambda(q)} a_i x_i + b$.
   • **Case** $a_{\lambda(q)} = 0$**.** Then $D = \texttt{comp}(D, \lambda(q)) \in E_{\lambda(q),preg}$, so we can decide whether $D \preceq_{\lambda(q)} 0$ and $g' \preceq'_k h'$ iff $D \preceq_{\lambda(q)} 0$.
   • **Case** $a_{\lambda(q)} \in \mathbb{Q} \setminus \{0\}$**.** Then $\texttt{compnorm}(D, \lambda(q)) \in E_{\lambda(q),preg}$. There are four subcases to consider. For instance if $a_{\lambda(q)} > 0$ and $x_{\lambda(q)} \preceq_{\lambda(q)} \texttt{compnorm}(D, \lambda(q))$ then $g' \preceq'_k h'$. The other subcases are similar.
   • **Case** $a_{\lambda(q)} \notin \mathbb{Q}$**.** Let us write $a_{\lambda(q)} = \frac{r_{\lambda(q)}}{s_{\lambda(q)}}$. According to the specification of *preg*, we know the sign of $a_{\lambda(q)}$ as $r_{\lambda(q)}$ belongs to *PolPar* and $s_{\lambda(q)}$ is a product of items in *PolPar*. In case $a_{\lambda(q)} = 0$, we decide $g' \preceq'_k h'$ as in the first case. Otherwise, we decide in a similar way as in the second case. For instance if $a_{\lambda(q)} > 0$ and $x_{\lambda(q)} \preceq_{\lambda(q)} \texttt{compnorm}(D, \lambda(q))$ then $g' \preceq'_k h'$.

Observation 1. Let $\pi \in (preg, \preceq_1)$ and $(q,v) \in [\![R]\!]_\pi$. If there exists a transition $(q,v) \xrightarrow{e} (q',v')$ then for some R', there is a transition $R \xrightarrow{e} R'$ in $\mathcal{R}(preg, \preceq_1)$ and (q',v') belongs to $[\![R']\!]_\pi$.

Conversely, if there is a transition $R \xrightarrow{e} R'$ in $\mathcal{R}(preg, \preceq_1)$ then for each $\pi \in (preg, \preceq_1)$ and each $(q,v) \in [\![R]\!]_\pi$ there exists $(q',v') \in [\![R']\!]_\pi$ such that $(q,v) \xrightarrow{e} (q',v')$.

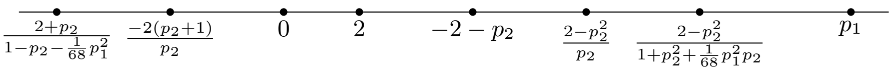
Time step. Let $R = (q, \{\preceq_k\}_{1 \leq k \leq \lambda(q)})$. There is a transition $R \xrightarrow{succ} Post(R)$ for $Post(R) = (q, \{\preceq'_k\}_{1 \leq k \leq \lambda(q)})$, the time successor of R, which is defined as follows.

For each $i < \lambda(q)$, $\preceq'_i = \preceq_i$. Now let $\sim$ be the equivalence relation $\preceq_{\lambda(q)} \cap \preceq^{-1}_{\lambda(q)}$ induced by the preorder. On equivalence classes, this (total) preorder becomes a (total) order. Let V be the equivalence class containing $x_{\lambda(q)}$.

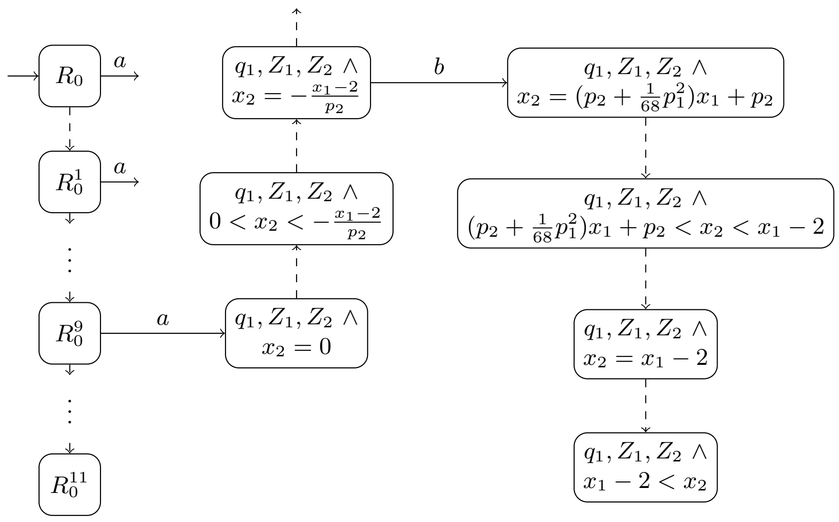
1. Either $V = \{x_{\lambda(q)}\}$ and it is the greatest equivalence class. Then $\preceq'_{\lambda(q)} = \preceq_{\lambda(q)}$ (thus $Post(R) = R$).
2. Either $V = \{x_{\lambda(q)}\}$ and it is not the greatest equivalence class. Let V' be the next equivalence class. Then $\preceq'_{\lambda(q)}$ is obtained by merging V and V', and preserving $\preceq_{\lambda(q)}$ elsewhere.
3. Either V is not a singleton. Then we split V into $V \setminus \{x_{\lambda(q)}\}$ and $\{x_{\lambda(q)}\}$ and "extend" $\preceq_{\lambda(q)}$ by $V \setminus \{x_{\lambda(q)}\} \preceq'_{\lambda(q)} \{x_{\lambda(q)}\}$.

Observation 2. Let $\pi \in (preg, \preceq_1)$ and $(q,v) \in [\![R]\!]_\pi$. There exists $d > 0$ such that $(q, v +_q d) \in [\![\, Post(R) \,]\!]_\pi$ and for each d' with $0 \leq d' \leq d$, $(q, v +_q d') \in [\![R]\!]_\pi \cup [\![Post(R)]\!]_\pi$.

Example 3. This construction is illustrated on automaton $\mathcal{A}_1$ of Fig. 1(a). We consider the parameter region $preg$ of $PolPar$ defined by:
$p_2 < 0$, $p_2 + 1 < 0$, $1 - p_2 - \frac{1}{68}p_1^2 > 0$ and $-1 - p_2^2 - \frac{1}{68}p_1^2 p_2 > 0$ and the ordering $\preceq_1$ of the expressions in $E_{1,preg} = E_1$ specified by the line below. Region $(preg, \preceq_1)$ is non empty since it includes parameter values $p_1 = 20$ and $p_2 = -5$.


$\frac{2+p_2}{1-p_2-\frac{1}{68}p_1^2}$
$\frac{-2(p_2+1)}{p_2}$
0
2
$-2-p_2$
$\frac{2-p_2^2}{p_2}$
$\frac{2-p_2^2}{1+p_2^2+\frac{1}{68}p_1^2 p_2}$
p_1

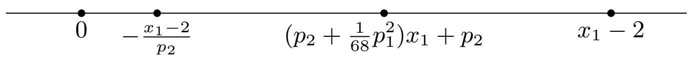
A part of the resulting class automaton $\mathcal{R}(preg, \preceq_1)$, including the trajectory in Fig. 1(b), is depicted in Fig. 2, where dashed lines indicate time steps. In state q_0, the only relevant clock is x_1 and the initial class is $R_0 = (q_0, Z_0)$ with Z_0 is $\preceq_1$ extended with $x_1 = 0$ Time successors of the initial state are obtained by moving x_1 to the right along the line:



R_0
a
$q_1, Z_1, Z_2 \wedge$ $x_2 = -\frac{x_1-2}{p_2}$
b
$q_1, Z_1, Z_2 \wedge$ $x_2 = (p_2 + \frac{1}{68}p_1^2)x_1 + p_2$
R_0^1
a
$q_1, Z_1, Z_2 \wedge$ $0 < x_2 < -\frac{x_1-2}{p_2}$
$q_1, Z_1, Z_2 \wedge$ $(p_2 + \frac{1}{68}p_1^2)x_1 + p_2 < x_2 < x_1 - 2$
R_0^9
a
$q_1, Z_1, Z_2 \wedge$ $x_2 = 0$
$q_1, Z_1, Z_2 \wedge$ $x_2 = x_1 - 2$
$q_1, Z_1, Z_2 \wedge$ $x_1 - 2 < x_2$
R_0^{11}

Fig. 2. An initial part of $\mathcal{R}(preg, \preceq_1)$ for $\mathcal{A}_1$

$R_0^1 = (q_0, \preceq_1 \wedge 0 < x_1 < 2)$, $R_0^2 = (q_0, \preceq_1 \wedge x_1 = 2)$, ..., up to $R_0^{11} = (q_0, \preceq_1 \wedge p_1 < x_1)$. Transition a can be fired from all these classes except from R_0^{10} and R_0^{11}. In Fig. 2, we represent only the one from R_0^9. The given region of $PolPar$ and the class from which a is fired determine the ordering of the expressions in $E_{2,preg} \setminus \{x_2\} = E_2 \setminus \{x_2\}$, specified by the line below. We denote by Z_1 the ordering $\preceq_1$ extended with $\frac{2-p_2^2}{1+p_2^2+\frac{1}{68}p_1^2p_2} < x_1 < p_1$ and Z_2 the ordering defined below. This firing produces class $R_1 = (q_1, (Z_1, Z_2 \wedge x_2 = 0))$. Transition b is fired from the (second) time successor of R_1 for which $x_2 = -\frac{x_1-2}{p_2}$.



Based on the decidability of first-order theory of the real numbers and the class automata construction, we obtain decidability for the considered reachability problems. When only additive parametrisation is considered, the existential reachability problem reduces to reachability for ITA, inducing a lower complexity.

Theorem 1. *The (scoped) existential, universal and robust reachability problems for PITA are decidable and belong to 2EXPSPACE and PSPACE when the number of clocks is fixed.*

Theorem 2. *The (polyhedral scoped) existential reachability problem is decidable for additively parametrised PITA, and belongs to 2EXPTIME and PTIME when the number of clocks and parameters is fixed.*

4 Conclusion

While seminal results on parametrised timed models leave little hope for decidability in the general case, we provide here an expressive formalism for the analysis of parametric reachability problems. Our setting includes a restricted form of stopwatches and polynomials in the parameters occurring as both additive and multiplicative coefficients of the clocks in guards and updates. We plan to investigate which kind of timed temporal logic would be decidable on PITA.

References

1. Alur, R., Henzinger, T.A., Vardi, M.Y.: Parametric real-time reasoning. In: Proc. of ACM Symp. on Theory of Computing, ACM (1993) 592–601
2. Alur, R., Dill, D.L.: Automata for modeling real-time systems. In: Proc. of ICALP'90, Springer (1990) 322–335
3. Bérard, B., Fribourg, L.: Automated verification of a parametric real-time program: The ABR conformance protocol. In: Proc. of CAV'99. Volume 1633 of LNCS., Springer (1999) 96–107
4. Miller, J.S.: Decidability and complexity results for timed automata and semi-linear hybrid automata. In: Proc. of HSCC 2000. Volume 1790 of LNCS. (2000) 296–309
5. Doyen, L.: Robust Parametric Reachability for Timed Automata. Information Processing Letters **102**(5) (2007) 208–213
6. André, É., Chatain, Th., Encrenaz, E., Fribourg, L.: An inverse method for parametric timed automata. Int. J. of Foundations of Comp. Sci. **20**(5) (2009) 819–836
7. André, É., Fribourg, L., Kühne, U., Soulat, R.: IMITATOR 2.5: A tool for analyzing robustness in scheduling problems. In: Proc. of FM'12. Volume 7436 of LNCS., Springer (2012) 33–36
8. Jovanović, A., Lime, D., Roux, O.H.: Integer parameter synthesis for timed automata. In: Proc. of TACAS'13. Volume 7795 of LNCS., Springer (2013) 391–405
9. Alur, R., Henzinger, T.A., Ho, P.H.: Automatic Symbolic Verification of Embedded Systems. IEEE Transactions on Software Engineering **22** (1996) 181–201
10. Henzinger, T.A., Ho, P.H., Wong-Toi, H.: HyTech: A Model-Checker for Hybrid Systems. Software Tools for Technology Transfer **1** (1997) 110–122
11. Bozzelli, L., Torre, S.L.: Decision problems for lower/upper bound parametric timed automata. Formal Methods in System Design **35**(2) (2009) 121–151
12. Hune, T., Romijn, J., Stoelinga, M., Vaandrager, F.: Linear parametric model checking of timed automata. J. of Logic and Alg. Prog. **52-53** (2002) 183–220
13. Jovanović, A., Faucou, S., Lime, D., Roux, O.H.: Real-time control with parametric timed reachability games. In: Proc. of WODES'12, IFAC (2012) 323–330
14. Bérard, B., Haddad, S.: Interrupt Timed Automata. In: Proc. of FoSSaCS'09. Volume 5504 of LNCS., York, GB, Springer (2009) 197–211
15. Bérard, B., Haddad, S., Sassolas, M.: Interrupt timed automata: Verification and expressiveness. Formal Methods in System Design **40**(1) (2012) 41–87
16. Canny, J.F.: Some algebraic and geometric computations in PSPACE. In: Twentieth ACM Symp. on Theory of Computing. (1988) 460–467

A Proofs

Lemma 2. *The construction procedure of $\{E_k\}_{k\leq n}$ terminates and the size of every E_k is bounded by $(2E+2)^{2^{n(n-k+1)}+1}$ where E is the number of atomic propositions in edges of the PITA.*

Proof. Given some k, we prove the termination of the stage relative to k. Observe that the step 2(b) only adds new expressions to $E_{k'}$ for $k' < k$. Thus the steps 2(a) and 2(b) can be ordered. Let us prove the termination of the step 2(a) of the saturation procedure. We define E_k^0 as the set E_k after step 1 and E_k^i as this set after insertion of the i^{th} item in step 2(a). With each added item $C[u]$ can be associated its *father* C. Thus we can view E_k as an increasing forest with finite degree (due to the finiteness of the edges) and finitely many roots. We claim that the length of any branch is at most 2^k. Let $C_0, C_1, \ldots$ be an arbitrary branch where $C_{i+1} = C_i[u_i]$ for some update u_i such that $C_{i+1} \neq C_i$. Observe that the number of updates that change the variable x_k is either 0 or 1 since once x_k disappears it cannot appear again. We split the branch into two parts before and after this update or we still consider the whole branch if there is no such update. In these (sub)branches, we conclude with the same reasoning that there is at most one update that change the variable x_{k-1}. Iterating this process, we conclude that the number of updates is at most $2^k - 1$ and the length of the branch is at most 2^k.

For the sake of readability, we set $B = 2E + 2$. The final size of E_k is thus at most $|E_k^0| \times B^{2^k}$ since the width of the forest is bounded by B.

In the second step, we add at most $B \times (|E_k| \times (|E_k| - 1))/2$ to E_i for every $i < k$. The final bound is obtained by an induction similar to the one for ITA [15]. □

Lemma 3. *Let c be an integer constant occurring in an expression of PolPar or some E_k. Then the number of bits required for c is at most $(n+2)!2^n b_0$ where b_0 is the maximal number of bits for integers occurring in the PITA.*

The maximal degree of polynomials occurring in these expressions is at most $(n+2)!d_0$ where d_0 is the maximal degree of polynomials occurring in the PITA.

Proof. Assume that before the level $n-k$ is performed, the number of bits of an integer occurring in some expression is b_k. We establish a relation between b_k and b_{k+1}. At level $n-k$, step 1 does not induce any increasing since it applies normalisation on guards. More precisely the numerators of

rational fractions are unchanged while the denominators are numerators of some previous expressions.
Let us examine an expression $C = \sum_{i \leq n-k} a_i x_i + b$ built after step 2(a). Examining the successive updates, the numerator of coefficient a_i can be expressed as $\sum_{d \in \mathcal{D}} \prod_{j \in d} c_{d,j}$ where $\mathcal{D}$ is the set of subsets of $\{i, \ldots, n-k\}$ containing i and $c_{d,j}$ are either coefficients of the updates or coefficients of an expression built before this step. The same reasoning applies to the numerator of b. So the number of bits of the coefficients of the rational functions a_i's and b is bounded by: $(n-k)(b_k+1)$. The denominators are denominators of expressions previously built.
At step 2(b), the difference $C[u] - C'[u]$ requires to compute the lcm of two denominators (bounded by their product). So the difference operation leads to a bound $2 + (n-k)(b_k+1)$ for the numerators of its coefficients and $2b_k$ for the denominators.
The final normalisation at step 2(b) consists in multiplying a numerator and a denominator of some coefficients leading to a bound $(n-k+2)(b_k+1)$. So $b_{k+1} \leq (n-k+2)(b_k+1) \leq (n-k+2)2b_k$ yielding the desired bound.

Assume that before the level $n-k$ is performed, the number of bits of an integer occurring in some expression is d_k. We establish a relation between d_k and d_{k+1}. At level $n-k$, step 1 does not induce any increasing since it applies normalisation on guards. More precisely the numerators of rational fractions are unchanged while the denominators are numerators of some previous expressions.
Let us examine an expression $C = \sum_{i \leq n-k} a_i x_i + b$ built after step 2(a). Examining the successive updates, the numerator of coefficient a_i can be expressed as $\sum_{d \in \mathcal{D}} \prod_{j \in d} c_{d,j}$ where $\mathcal{D}$ is the set of subsets of $\{i, \ldots, n-k\}$ containing i and $c_{d,j}$ are either coefficients of the updates or coefficients of an expression built before this step. The same reasoning applies to the numerator of b. So the degree of the a_i's and b is bounded by: $(n-k)b_k$. The denominators are denominators of expressions previously built.
At step 2(b), the difference $C[u] - C'[u]$ requires to compute the lcm of two denominators (bounded by their product). So the difference operation leads to a bound $(n-k)b_k$ for the numerators of its coefficients and $2b_k$ for the denominators.
The final normalisation at step 2(b) consists in multiplying a numerator and a denominator of some coefficients leading to a bound $(n-k+2)b_k$. So $b_{k+1} \leq (n-k+2)b_k$ yielding the desired bound. $\square$

Proposition 1 (Soundness and completeness). *Let $(preg, \preceq_1)$ be a non empty parameter region of a PITA $\mathcal{A}$ and $\pi \in (preg, \preceq_1)$. Then:*

– *For any class R of automaton $\mathcal{R}(preg, \preceq_1)$ there exists a configuration (q, v) reachable from $(q_0, \mathbf{0})$ in $\mathcal{T}_{\mathcal{A},\pi}$, such that $(q, v) \in [\![R]\!]_\pi$.*
– *For any (q, v) reachable from $(q_0, \mathbf{0})$ in $\mathcal{T}_{\mathcal{A},\pi}$, there exists a class R of automaton $\mathcal{R}(preg, \preceq_1)$ such that $(q, v) \in [\![R]\!]_\pi$.*

Proof. Using observations 1 and 2 related to the construction above, we directly obtain the given proposition.

Theorem 1. *The (scoped) existential, universal and robust reachability problems for PITA are decidable and belong to 2EXPSPACE and PSPACE when the number of clocks is fixed.*

Proof. Using lemma 2 and proposition 1 we design a non deterministic procedure for existential reachability of q:

1. Build $PolPar$ and $\{E_k\}_{1 \leq k \leq n}$.
2. Guess a parameter region $(preg, \preceq_1)$.
3. Check non emptiness of $(preg, \preceq_1)$.
4. Build the class automaton $\mathcal{R}(preg, \preceq_1)$ and check whether q occurs in some class.

For universal reachability of q in step 4, one checks whether q does not occur in any class. This gives us a non deterministic procedure for the complementary problem. For robust reachability in step 2, one guesses an open parameter region i.e. only specified by strict inequalities.
Let us analyse the complexity of these procedures. Due to lemmas 2 and 3, the first step is performed in 2EXPTIME and in PTIME when the number of clocks is fixed. Guessing has the same complexity.
The satisfiability problem for a first-order formula is in PSPACE [16]. Let s be the number of (in)equalities specifying the region, due to lemma 2, $s = (2E)^{2^{O(n^2)}}$ where E is the number of atomic propositions in edges of the PITA. Let b be the maximal number of bits of an integer occurring in the specification of the region, due to lemma 3, $b = 2^{O(n \log(n)} b_0$ where b_0 is the maximal number of bits for an integer occurring in the PITA. Let d be the maximal degree of the polynomials occurring in the specification of the region due to the same lemma, $d = 2^{O(n \log(n)} d_0$ where d_0 is the maximal degree for a polynomial occurring in the PITA. So the emptiness problem for a region is performed in 2EXPSPACE which becomes PSPACE when the number of clocks is fixed.
Observe now that the class automaton $\mathcal{R}(preg, \preceq_1)$ is isomorphic to the class automaton of the ITA that would be obtained by applying any parameter valuation in $(preg, \preceq_1)$. It has been proved in [14] that this

automaton can be built in polynomial time w.r.t. the size of the representation of any class. As the size of the representation of a class of a PITA has the same order as the one of the corresponding ITA (dominated by the doubly exponential number of expressions) and the construction algorithms perform similar operations, this yields a complexity of 2EXPTIME and PTIME when the number of clocks is fixed.
So the dominating factor of this non deterministic procedure is the emptiness check done in 2EXPSPACE. By Savitch theorem this procedure can be determinised with the same complexity.

□

Proposition 2. *For every additively parametrised PITA $\mathcal{A}$, with states Q and initial state q_0, there exist a (non-parametric) ITA $\mathcal{A}'$, with states Q', containing Q, and initial state q'_0 fulfilling the following equivalence. For every $q \in Q$:*

there exists π such that q is reachable from q_0 in $\mathcal{A}$ for π
iff q is reachable from q'_0 in $\mathcal{A}'$

Proof. We propose a transformation of an additively parametrised PITA $\mathcal{A}$ with n clocks (and thus n levels), and k parameters $p_1, ..., p_k$, into an equivalent ITA $\mathcal{A}'$ with $n + k + 1$ levels. This transformation is a way to reduce the parametric reachability problem of additively parametrised PITA to the reachability problem of ITA, that is known to be decidable [15], and the construction is shown in Fig. 3.

ITA $\mathcal{A}'$ consists of a prefix (the first $k + 1$ levels) and the original automaton $\mathcal{A}$ (n levels), where we put clocks in the place of parameters. To that end, we introduce k new clocks, for simplicity we also call them $p_1, ..., p_k$, and new auxiliary clock p_0. Each clock p_i is active in level $i + 1$ in (the prefix of) $\mathcal{A}'$.

In the first level of $\mathcal{A}'$, clock p_0 is active. After some (arbitrary) time, a transition, with no guard, is taken to the state of the second level and clock p_0 is stopped. In the second level, clock p_1 is active and the same procedure continues: after some time a transition to the next level is taken, and clock p_1 is stopped, and so on for the first k levels. Level $k+1$ consists of k states and a clock p_k is active. In the first k levels we have chosen the absolute value for the clocks (parameters in $\mathcal{A}$), and level $k + 1$ serves to choose the final sign of clocks, which is done by assigning p_{i-1} or $-p_{i-1}$ to clock p_i, between each two consecutive states, for all $i \in [1..k - 1]$, in a run without a delay in any of the states of level $k+1$ (the other runs, with delays in those states, overlap on those corresponding to other parameter valuations and are therefore not a problem). In the last state of level $k+1$, the stopped clocks $p_1, ..., p_k$ can therefore have any arbitrary real value assigned. The automaton immediately proceeds to the initial state of $\mathcal{A}$ keeping the values of clocks. The obtained automaton $\mathcal{A}'$ is an ITA, since parameters of $\mathcal{A}$ are modeled as clocks in $\mathcal{A}'$, for which the reachability problem is decidable.

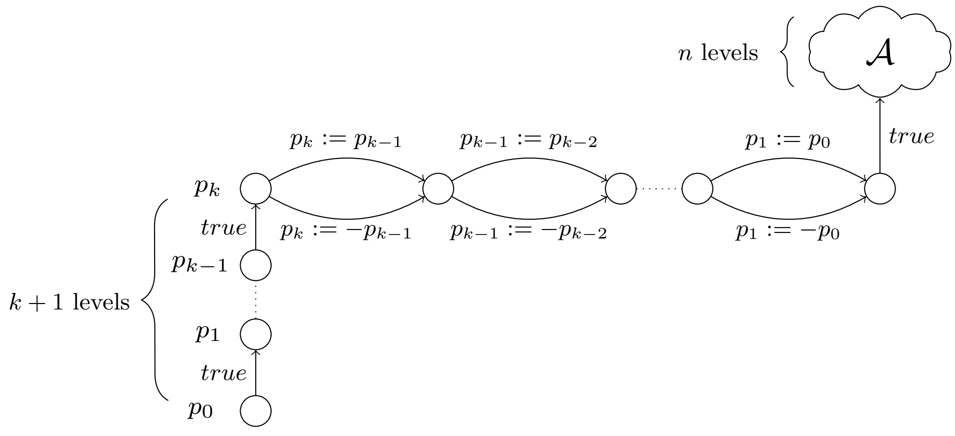


Fig. 3. An equivalent ITA $\mathcal{A}'$

Let X be the set of clocks in $\mathcal{A}$ and X' be the set of clocks in $\mathcal{A}'$ (thus $X' = X \cup \{p_0, ..., p_k\}$). For any subset $Y \subseteq X$ and a valuation v, we define the restriction of v to Y as the unique valuation v on Y such that $v_{|Y}(x) = v(x)$. Now will now show that a configuration $s = (q, v)$ is reachable in $\mathcal{A}$ for π (which we note $\mathcal{A}_\pi$) iff the corresponding configuration $s' = (q', v')$, such that $q' = q, \forall x \in X, v'_{|X}(x) = v(x)$, is reachable in $\mathcal{A}'$.

We first state that for all $(r_1, ..., r_k) \in \mathbb{R}^k$, there exists a run in $\mathcal{A}'$ from the initial configuration of $\mathcal{A}'$ to a configuration (q_0, v), such that $(q_0, v_{|X})$ is the initial configuration of $\mathcal{A}_\pi$ and $\forall i, v_{|X' \setminus X}(p_i) = \pi(p_i) = r_i$. Then, if there exists a path to reach s' in $\mathcal{A}'$, then there exists a parameter valuation π such that the state s is reachable in $\mathcal{A}_\pi$. A value for each parameter p_i in $\mathcal{A}_\pi$ corresponds to the value of the clock p_i in $\mathcal{A}'$ assigned in the level $k+1$. On the other hand, if there is no path to reach s' in $\mathcal{A}'$ then for all the possible prefixes corresponding to the first $k+1$ levels of a path, the possible continuations never lead to s, which is equivalent to for all parameter valuations π, the related configuration in $\mathcal{A}_\pi$ is not reachable in $\mathcal{A}_\pi$. This, in turn, means that the set of parameter valuations is empty. This shows that the two problems are indeed equivalent. $\square$

Theorem 2. *The (polyhedral scoped) existential reachability problem is decidable for additively parametrised PITA, and belongs to 2EXPTIME and PTIME when the number of clocks and parameters is fixed.*

Proof. Following Proposition 2, every additively parametrised PITA can be transformed into an equivalent ITA, and the (unscoped) reachability problem of additively parametrised PITA is thus reduced to the reachability problem of ITA, already known to be decidable. The complexity results follow from the complexity results of ITA [15], since the size of $\mathcal{A}'$ is only linear in the size $\mathcal{A}$: if there are n clocks, k parameters, x states and y transitions in $\mathcal{A}$, the number of clocks, states and transitions in $\mathcal{A}'$ is $n + k + 1$, $x + 2k + 1$ and $y + 3k + 1$, respectively.

With a polyhedral scope, given as a finite union of polyhedra, we need to guard the transition between the last state of the prefix and the initial state of $\mathcal{A}$, in $\mathcal{A}'$, by the given polyhedra (each polyhedra of the union could guard a different transition, as well). $\square$