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 to enrich ITA with parameters in the spirit above. A PITA is 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 F, G, we denote by $\mathcal{P}ol(F, G)$, the set of polynomials with variables in F and coefficients in G. We also denote

by $\mathcal{L}in(F, G)$ the subset of polynomials with degree at most one and by $\mathcal{F}rac(F, G)$, the set of rational functions with variables in F and coefficients in G (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 in $\mathcal{L}in(X, \mathcal{P}ol(P, \mathbb{Q}))$ and $\bowtie \in \{>, \geq, =, \leq, <\}$.

Updates and valuations. An *update* is a conjunction of assignments of the form $\wedge_{x \in X} x := C_x$, where $C_x \in \mathcal{L}in(X, \mathcal{P}ol(P, \mathbb{Q}))$, with possibly $C_x = 0$ or $C_x = x$. 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.*, simultaneously substituting each x by C_x in C, if $x := C_x$ is the update for x in u. For instance, for clocks $X = \{x_1, x_2\}$, parameters $P = \{p_1, p_2, p_3\}$, expression $C = p_2 x_2 - 2x_1 + 3p_1$ and the update u defined by $x_1 := 1 \wedge x_2 := p_3 x_1 + p_2$, applying u to C yields the expression $C[u] = p_2 p_3 x_1 + p_2^2 + 3p_1 - 2$. Note that the use of multiplicative parameters for clocks may result in polynomial coefficients when updates are applied.

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 $x_1 := 1 \wedge x_2 := x_2 \wedge x_3 := p_1 x_3 - x_1$, 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 a finite set of transitions. Let* $q \xrightarrow{\varphi, a, u} q'$ *be a transition in* Δ *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.

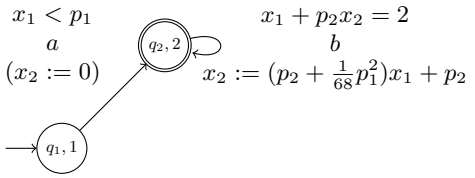
Definition 2. *The semantics of a PITA* $\mathcal{A}$ *w.r.t. a parameter valuation* π *is defined by the (timed) transition system* $\mathcal{T}_{\mathcal{A},\pi} = (S, s_0, \to)$*. The set of configurations is* $S = \{(q, v) \mid q \in Q,\ v \in \mathbb{R}^X\}$*, with initial configuration* $s_0 = (q_0, \mathbf{0})$*. 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 $v' = v +_q d$.*

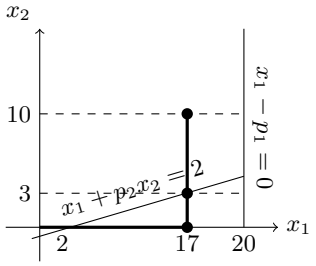
Discrete steps: *A discrete step $(q, v) \xrightarrow{e} (q', v')$ can occur for some transition $e = q \xrightarrow{\varphi,a,u} q'$ in Δ such that $\pi, v \models \varphi$ and $v'(x) = \pi(v[u](x))$.*

A *run* of $\mathcal{A}$ for some parameter valuation π is a finite path in the transition system $\mathcal{T}_{\mathcal{A},\pi}$, which can be written as an alternating sequence of (possibly null) time and discrete steps. 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 $\mathcal{A}$ is depicted in Fig. 1(a), with two interrupt levels. Fixing the parameter valuation π: $p_1 = 20$ and $p_2 = -5$, the run $(q_1, 0, 0) \xrightarrow{17} (q_1, 17, 0) \xrightarrow{a} (q_2, 17, 0) \xrightarrow{3} (q_1, 17, 3) \xrightarrow{b} (q_2, 17, 10)$ is obtained as follows. After staying in q_1 for 17 time units, a can be fired and the value of x_1 is then frozen in state q_2, while x_2 increases. Transition b can be taken if $x_1 + p_2 x_2 = 2$, hence for $x_2 = 3$, after which x_2 is updated to $x_2 = 18p_2 + \frac{17}{68}p_1^2 = 10$. A geometric view of this run w.r.t. π is given (in bold) in Fig. 1(b).



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



(b) A possible run in $\mathcal{A}$ for π

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 a first order formula over 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' (where $\|\pi\|_\infty = max_{p \in P}|\pi(p)|$). When satisfied, this property

ensures that small parameter perturbations do not modify the reachability result. It is also related to parameter synthesis where a valuation has to be enlarged to an open region with the same reachability goal.

3 Reachability Analysis

In this section, we give the main construction for the decidability result (Point 1 below), the remaining part of the proof is given in appendix.

Theorem 1. *1. The (scoped) existential, universal and robust reachability problems for PITA are decidable and belong to 2EXPSPACE. The complexity reduces to PSPACE when the number of clocks is fixed.*
2. The (polyhedral scoped) existential reachability problem is decidable for additively parametrised PITA, and belongs to 2EXPTIME. It belongs to PTIME when the number of clocks and parameters is fixed.

We briefly present the main ideas underlying the proof. Given a PITA $\mathcal{A}$, the first step is to build a finite partition of the set $\mathbb{R}^P$ of parameter valuations. An element Π of this partition is specified by a satisfiable first-order formula over $(\mathbb{R}, +, \times)$, with the parameters as variables. Intuitively, inside Π the qualitative behaviour of $\mathcal{A}$ does not depend on the precise parameter valuation. In a second step, we build a finite automaton $\mathcal{R}(\Pi)$ for each non empty Π. In $\mathcal{R}(\Pi)$, a state R, called a *class*, defines a set $[\![R]\!]_\pi$ of reachable configurations of $\mathcal{T}_{\mathcal{A},\pi}$ for a valuation $\pi \in \Pi$. The transition relation of $\mathcal{R}(\Pi)$ contains discrete steps $R \xrightarrow{e} R'$ (for a transition e of $\mathcal{A}$) and abstract time steps $R \to Post(R)$ with the following properties:

Discrete Step (DS): If there is a transition $R \xrightarrow{e} R'$ in $\mathcal{R}(\Pi)$ then for each $\pi \in \Pi$ and each $(q, v) \in [\![R]\!]_\pi$ there exists $(q', v') \in [\![R']\!]_\pi$ such that $(q, v) \xrightarrow{e} (q', v')$.
Conversely, let $\pi \in \Pi$ and $(q, v) \in [\![R]\!]_\pi$. If there exists a transition $(q, v) \xrightarrow{e} (q', v')$ in $\mathcal{T}_{\mathcal{A},\pi}$ then for some R', there is a transition $R \xrightarrow{e} R'$ in $\mathcal{R}(\Pi)$ and (q', v') belongs to $[\![R']\!]_\pi$.

Time Step (TS): Let $\pi \in \Pi$ 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 $0 \leq d' \leq d$, $(q, v +_q d') \in [\![R]\!]_\pi \cup [\![Post(R)]\!]_\pi$.

Hence, we obtain a finite family of abstract time bisimulations of the transition systems $\mathcal{T}_{\mathcal{A},\pi}$, for all parameter valuations, which gives the decidability result.

The key idea for the construction of $\mathcal{R}(\Pi)$ is based on the fact that, at some level k, the active clock x_k evolves in a one dimensional space

and must be compared to a set E_k of expressions, the values of which are based on parameter values and the (fixed) clock values of levels below. For instance, in the automaton of Fig. 1(a), if $p_2 = 0$, the guard reduces to a comparison of $x_1 - 2$ with 0. If $p_2 \neq 0$, clock x_2 must be compared to $-\frac{x_1-2}{p_2}$ (in a sense depending on the sign of p_2 if the constraint was an inequality). After transition b is fired, updates must also be taken into account which leads to enlarge the set of expressions. Due to the syntactic restrictions of PITA this procedure terminates. Hence, we first need to define a set $PolPar$ of polynomials (appearing in the denominators like p_2) and a family $\{E_k\}_{k\leq n}$ of expressions in $\mathcal{L}in(X_k, \mathcal{F}rac(P, \mathbb{Q}))$.

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

We define operations on expressions, relatively to a level k, to help building the elements in E_k to which the clock x_k will be compared.

Definition 3. *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 $\mathcal{P}ol(P, \mathbb{Q})$. We associate with C 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* $\texttt{compnorm}(C, k) = -\sum_{i<k} \frac{a_i}{a_k} x_i - \frac{b}{a_k}$.

In the previous example, `comp` corresponds to $x_1 - 2$ while `compnorm` corresponds to $-\frac{x_1-2}{p_2}$. More examples are given after the construction of $PolPar$ and $\{E_k\}_{k\leq n}$. This construction proceeds top down from level n to level 1 after initialising $PolPar$ to $\emptyset$ and E_k to $\{x_k, 0\}$ 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 $\{C, C'\}$ be a set of two expressions in E_k. We compute $C'' = C[u] - C'[u]$, choosing an arbitrary order between C and C'. This step ends by handling 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 $E_2 = \{x_2, 0\}$. Starting with level $k = 2$, we consider in step 1 the expression $C_2 = p_2x_2 + x_1 - 2$ appearing in the guard of the single edge leaving q_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 $E_2 = \{x_2, 0, x_1 - 2, -\frac{x_1-2}{p_2}\}$. For step 2(a) and the same edge, we apply its update to the expressions of E_2 that contain x_2, add them to E_2, and thus obtain $E_2 = \{x_2, 0, x_1 - 2, -\frac{x_1-2}{p_2}, (p_2 + \frac{1}{68}p_1^2)x_1 + p_2\}$.

In step 2(b), considering the single edge from q_1 to q_2, we compute the differences between any two expressions from E_2 (after applying update) and the resulting expressions `lead`, `comp` and `compnorm`, 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 proceed with level 1 and add $\texttt{compnorm}(C_1, 1) = p_1$ to E_1, hence:
$E_1 = \{x_1, 0, 2, -\frac{2(p_2+1)}{p_2}, -2 - p_2, \frac{2+p_2}{1-p_2-\frac{1}{68}p_1^2}, \frac{2-p_2^2}{p_2}, \frac{2-p_2^2}{1+p_2^2+\frac{1}{68}p_1^2p_2}, p_1\}$,
$E_2 = \{x_2, 0, x_1 - 2, -\frac{x_1-2}{p_2}, (p_2 + \frac{1}{68}p_1^2)x_1 + p_2\}$ 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. This algorithm manipulates rationals numbers (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 there exists polynomials $P_1, \ldots, P_\ell \in PolPar$ and some constant $K \in \mathbb{Q} \setminus \{0\}$ such that $s = K.\prod_{1\leq i\leq \ell} P_i$.*

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 $\mathcal{A}$ be a PITA, and let b_0 be the maximal number of bits for integers and d_0 the maximal degree of polynomials, occurring in $\mathcal{A}$. If b is the number of bits of an integer constant and d is the degree of a polynomial, occurring in an expression of $PolPar$ or some E_k, then $b \leq (n+2)!2^n b_0$ and $d \leq (n+2)!d_0$.*

We now explain the partition construction. Starting from the finite set $PolPar$, we split the set of parameter valuations in parameter regions

specified by the result of comparisons to 0 of the values of the polynomials in $PolPar$. For instance, for the set $PolPar$ computed above, the inequalities $p_2 < 0$, $p_2 + 1 < 0$, $1 - p_2 - \frac{1}{68}p_1^2 > 0$ and $-1 - p_2^2 - \frac{1}{68}p_1^2 p_2 > 0$ define a set $preg$ of parameter valuations containing $p_1 = 20$ and $p_2 = -5$. The set of non empty such regions can be computed by solving an existential formula of the first-order theory of reals.

Then, 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 $\mathcal{F}rac(P, \mathbb{Q})$ and that, depending on the parameter valuation, two different expressions can produce the same value. We refine $preg$ according to a linear pre-order $\preceq_1$ on $E_{1,preg} \setminus \{x_1\}$ which is satisfiable within $preg$. We denote this refined region by $\Pi = (preg, \preceq_1)$ and we now build a finite automaton $\mathcal{R}(\Pi)$.

3.2 Construction of the class automata

In this paragraph, we fix a non empty parameter region $\Pi = (preg, \preceq_1)$.

Class definition. A state of $\mathcal{R}(\Pi)$, called 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 $1 \leq k \leq \lambda(q)$. For a parameter valuation $\pi \in \Pi$, the class R describes the following subset of configurations in $\mathcal{T}_{\mathcal{A},\pi}$:
$[\![R]\!]_\pi = \{(q, v) \mid \forall k \leq \lambda(q)\ \forall g, h \in E_{k,preg},\ \pi(v(g)) \leq \pi(v(h)) \text{ iff } g \preceq_k h\}$

The initial state of $\mathcal{R}(\Pi)$ is the class R_0, such that $(q_0, \mathbf{0}) \in [\![R_0]\!]_\pi$, which can be straightforwardly determined by extending $\preceq_1$ to $E_{1,preg}$ with $x_1 \preceq_1 0$ and $0 \preceq_1 x_1$ and closing $\preceq_1$ by transitivity.

As usual, transitions in $\mathcal{R}(\Pi)$ consist of discrete 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 $\pi \in \Pi$, 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 $\pi \in \Pi$ 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 $C_j = \mathtt{comp}(C_j, \lambda(q)) \in E_{\lambda(q),preg}$ 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** $a_{\lambda(q)} \in \mathbb{Q}\setminus\{0\}$**.** 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 R' definition. Let $k \leq \lambda(q')$ and $g', h' \in E_{k,preg}$.

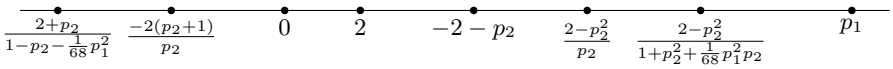
1. Either $k \leq \lambda(q)$, by step 2(a) of the construction, $g'[u], h'[u] \in E_{k,preg}$. Then $g' \preceq'_k h'$ iff $g'[u] \preceq_k h'[u]$.
2. Or $k > \lambda(q)$, let $D = g'[u] - h'[u] = \sum_{i\leq\lambda(q)} a_i x_i + b$.
   • **Case** $a_{\lambda(q)} = 0$**.** Then $D = \texttt{comp}(D, \lambda(q)) \in E_{\lambda(q),preg}$, 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 $g' \preceq'_k h'$.

Time step. For $R = (q, \{\preceq_k\}_{1\leq k\leq\lambda(q)})$, there is a transition $R \xrightarrow{succ} Post(R)$, where $Post(R) = (q, \{\preceq'_k\}_{1\leq k\leq\lambda(q)})$ is the time successor of R, defined as follows. Intuitively, all preorders below $\lambda(q)$ are fixed, so $\preceq'_i = \preceq_i$ for each $i < \lambda(q)$. On level $\lambda(q)$, the clock value simply progresses along the one dimensional time line, where the expressions are ordered. More precisely, let $\sim$ be the equivalence relation $\preceq_{\lambda(q)} \cap \preceq^{-1}_{\lambda(q)}$ induced by the preorder. A $\sim$-equivalence class groups expressions yielding the same value, and on these classes, the (total) preorder becomes a (total) order. Let V be the $\sim$-equivalence class containing $x_{\lambda(q)}$.

1. Either $V = \{x_{\lambda(q)}\}$. If V is the greatest $\sim$-equivalence class, then $\preceq'_{\lambda(q)} = \preceq_{\lambda(q)}$ (and $Post(R) = R$). Otherwise, let V' be the next $\sim$-equivalence class. Then $\preceq'_{\lambda(q)}$ is obtained by merging V and V', and preserving $\preceq_{\lambda(q)}$ elsewhere.
2. Or 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 $V \setminus \{x_{\lambda(q)}\} \preceq'_{\lambda(q)} \{x_{\lambda(q)}\}$.

Example 3. This construction is illustrated on automaton $\mathcal{A}_1$ of Fig. 1(a), for the region $\Pi = (preg, \preceq_1)$, where $preg$ was defined above 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 $\preceq_1$ is the ordering of the expressions in $E_{1,preg} = E_1$ specified by the line below.



A part of the resulting class automaton $\mathcal{R}(\Pi)$, including the run corresponding to the one in Fig. 1(b), is depicted in Fig. 2, where dashed lines indicate (abstract) time steps. The initial class is $R_0 = (q_0, Z_0)$ where 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: $R_0^1 = (q_0, \preceq_1 \wedge\ 0 < x_1 < 2)$, $R_0^2 = (q_0, \preceq_1 \wedge\ x_1 = 2)$, ..., up to $R_0^{11} = (q_0, \preceq_1 \wedge\ p_1 < x_1)$.

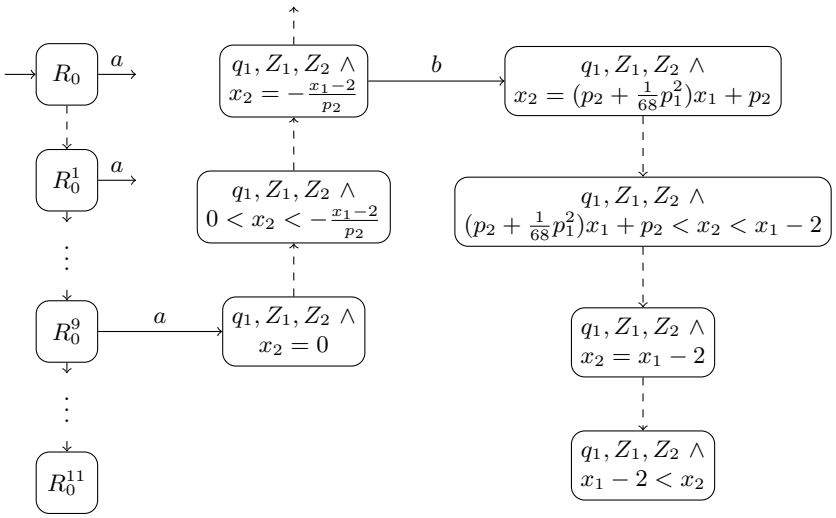
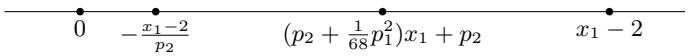


Fig. 2. A part of $\mathcal{R}(\Pi)$ for $\mathcal{A}_1$

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, and we denote by Z_1 the ordering $\preceq_1$ extended with $\frac{2-p_2^2}{1+p_2^2+\frac{1}{68}p_1^2p_2} < x_1 < p_1$. Region Π and Z_1 determine the ordering $Z_2 = \preceq_2$ on $E_{2,preg} \setminus \{x_2\} = E_2 \setminus \{x_2\}$, specified by the line below. This firing produces $R_1 = (q_1, Z_1, Z_2 \wedge x_2 = 0))$. Transition b is fired from the time (second) successor of R_1 for which $x_2 = -\frac{x_1-2}{p_2}$.



To conclude, observe that the automaton $\mathcal{R}(\Pi)$ defined above has the properties **(DS)** and **(TS)** mentionned previously, and is hence a finite time abstract bisimulation of $\mathcal{T}_{\mathcal{A},\pi}$, for all parameter valuations $\pi \in \Pi$.

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: ACM Symp. on Theory of Computing, ACM (1993) 592–601
2. Alur, R., Dill, D.L.: Automata for modeling real-time systems. In: ICALP'90, Springer (1990) 322–335
3. Bérard, B., Fribourg, L.: Automated verification of a parametric real-time program: The ABR conformance protocol. In: 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: HSCC'00. Volume 1790 of LNCS., Springer (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: 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: 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: WODES'12, IFAC (2012) 323–330
14. Bérard, B., Haddad, S.: Interrupt Timed Automata. In: FoSSaCS'09. Volume 5504 of LNCS., 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