Timed automata with parametric updates

Étienne André
Université Paris 13, LIPN, CNRS, UMR 7030, F-93430
Villetaneuse, France

Didier Lime
École Centrale de Nantes, LS2N, CNRS, UMR 6004,
Nantes, France

Mathias Ramparison
Université Paris 13, LIPN, CNRS, UMR 7030, F-93430
Villetaneuse, France

***Abstract*—Timed automata (TAs) represent a powerful formalism to model and verify systems where concurrency is mixed with hard timing constraints. However, they can seem limited when dealing with uncertain or unknown timing constants. Several parametric extensions were proposed in the literature, and the vast majority of them leads to the undecidability of the EF-emptiness problem: "is the set of valuations for which a given location is reachable empty?" Here, we study an extension of TAs where clocks can be updated to a parameter. While the EF-emptiness problem is undecidable for rational-valued parameters, it becomes PSPACE-complete for integer-valued parameters. In addition, exact synthesis of the parameter valuations set can be achieved. We also extend these two results to the EF-universality ("are all valuations such that a given location is reachable?"), AF-emptiness ("is the set of valuations for which a given location is unavoidable empty?") and AF-universality ("are all valuations such that a given location is unavoidable?") problems.**

I. Introduction

Timed automata (TAs) [AD94] represent a powerful formalism to model and verify systems where concurrency is mixed with hard timing constraints. TAs are an extension of finite-state automata with clocks, *i. e.,* real-valued variables, that can be compared to integer constants and updated to 0 along edges (called reset in the literature). TAs benefit from many decidability results such as the reachability of a discrete location (and some undecidability results too, such as language inclusion).

Although TAs seem to be able to model many interesting problems related to timed concurrent systems, several extensions were studied. For instance, TAs where clocks can be updated to integer constants have been introduced in [BDFP04] and interesting decidability results have been obtained, depending amongst other restrictions of the nature of the clock constraints (*e. g.,* diagonal-free, *i. e.,* whether clocks are compared to each other) and the updates of clocks (*e. g.,* whether it is allowed to update a clock to its current value increased by some rational constant). In a different direction, stopping the time elapsing of at least one clock in a TA gives *stopwatch automata*, for which the reachability problem becomes undecidable [CL00].

Timed automata may turn inappropriate to verify systems where the timing constants are subject to some uncertainty, where they can range in intervals, or when they are simply not known at some early design stage. Extending timed automata with parameters in guards in place of integers gives *parametric timed automata* (PTAs) [AHV93] and alleviates this drawback by allowing parameters (unknown constants) in the timing constraints. In the PTA literature, the main problem studied is the reachability emptiness, or EF-emptiness ("is the set of timing parameter valuations for which a given location is reachable empty?"): it is "robustly" undecidable in the sense that, even when varying the setting, undecidability is preserved. For example, EF-emptiness is undecidable even for a single bounded parameter [Mil00], even for a single rational-valued or integer-valued parameter [BBLS15], even with only one clock compared to parameters [Mil00], or with strict constraints only [Doy07]. More generally, all non-trivial problems are undecidable for PTAs (see [And17] for a survey). The unavoidability emptiness where we seek for valuations for which some location will *always eventually* be reached, or AF-emptiness ("is the set of timing parameter valuations such that a given location is unavoidable empty?") is also undecidable [JLR15]. Similarly EF-universality and AF-universality ("are all timing parameter valuations such that…") are undecidable [ALR16a] for the general class of PTAs, while decidability results have been shown for L/U-PTAs [HRSV02], [BL09], [AL17]. Formal definitions of these problems are given in Section III-C.

Contribution: We show that extending timed automata with parametric updates, *i. e.,* the ability to update a clock to an unknown rational constant, leads to the undecidability of the four following problems: EF-emptiness, AF-emptiness, EF-universality, AF-universality. That is, it is undecidable to determine:

- whether the set of parameter valuations for which a run leads to a given location is empty;
- whether for all parameter valuations there is a run that leads to a given location;
- whether the set of parameter valuations for which a given location is unavoidable empty;
- whether for all parameter valuations a given location is unavoidable.

In contrast, when we restrict the parameters domain to integers, all four problems do not only become decidable, but we can achieve exact synthesis, *i. e.,* represent the full set of valuations for which a run or all runs lead(s) to a given

This work is partially supported by the ANR national research program PACS (ANR-14-CE28-0002). This is the author version of the manuscript of the same name published in the proceedings of the 18th International Conference on Application of Concurrency to System Design (ACSD 2018). The final version is available at dx.doi.org/10.1109/ACSD.2018.000-2.

location.

On the one hand, our undecidability results adds to the long list of undecidable parametric extensions of timed automata.

On the other hand, our decidability result enriches the notably short list of decidable such parametric extensions: the exact synthesis of integer-valued parameters compared as upper-bounds to clocks can be achieved [BL09]; the emptiness of the valuations set for which a location is reachable is decidable both for rational-valued *L/U-PTAs* (in which parameters are always either upper bounds or lower bounds) [HRSV02], and for rational-valued *integer-point PTAs*, a semantic class for which the membership is however undecidable (although we exhibited a syntactic subclass, namely *reset-PTAs*) [ALR16a]. And AF-universality is decidable for L/U-PTAs only if the parameters are bounded with closed bounds (*i. e.,* of the form $p \in [a, b]$). In the three latter cases (*i. e.,* L/U-PTAs and integer-point PTAs), exact synthesis cannot be achieved though [JLR15], [ALR16a], which makes our synthesis result a rarity, together with only [BL09].

Finally, our formalism is supported by the parametric model checker IMITATOR [AFKS12].

Outline: Section II recalls necessary definitions. Section III introduces our formalism of update-to-parameter timed automata. Section IV proves our general undecidability result, while Section V proves the decidability when parameters become integer-valued. Section VI concludes the article and outlines future research directions.

II. Preliminaries

Let $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}_+$ and $\mathbb{R}_+$ denote the sets of non-negative integers, integers, non-negative rational numbers and non-negative real numbers respectively.

Throughout this paper, we assume a set $\mathbb{X} = \{x_1, \ldots, x_H\}$ of *clocks*, *i. e.,* real-valued variables that evolve at the same rate. A clock valuation is a function $w : \mathbb{X} \to \mathbb{R}_+$. We identify a clock valuation w with the point $(w(x_1), \ldots, w(x_H))$ of $\mathbb{R}_+^H$. We write $\vec{0}$ for the clock valuation that assigns 0 to all clocks. Given $d \in \mathbb{R}_+$, $w + d$ denotes the valuation such that $(w + d)(x) = w(x) + d$, for all $x \in \mathbb{X}$.

We assume a set $\mathbb{P} = \{p_1, \ldots, p_M\}$ of *parameters*, *i. e.,* unknown constants. A parameter *valuation* v is a function $v : \mathbb{P} \to \mathbb{Q}_+$. An *integer* parameter valuation is a valuation $v : \mathbb{P} \to \mathbb{N}$. We identify a valuation v with the point $(v(p_1), \ldots, v(p_M))$ of $\mathbb{Q}_+^M$.

In the following, we assume $\bowtie \in \{<, \leq, \geq, >\}$.

A *parametric guard* g is a constraint over $\mathbb{X} \cup \mathbb{P}$ defined by inequalities of the form $x \bowtie z$, where z is either a parameter or a constant in $\mathbb{Z}$. A *non-parametric guard* is a parametric guard without parameters (*i. e.,* over $\mathbb{X}$).

Given a parameter valuation v, $v(g)$ denotes the constraint over $\mathbb{X}$ obtained by replacing in g each parameter p with $v(p)$. Likewise, given a clock valuation w, $w(v(g))$ denotes the expression obtained by replacing in $v(g)$ each clock x with $w(x)$. A clock valuation w *satisfies* constraint $v(g)$ (denoted by $w \models v(g)$) if $w(v(g))$ evaluates to true. We say that v *satisfies* g, denoted by $v \models g$, if the set of clock valuations satisfying $v(g)$ is nonempty. We say that g is *satisfiable* if $\exists w, v$ s.t. $w \models v(g)$.

A parametric update is a partial function $r : \mathbb{X} \rightharpoonup \mathbb{N} \cup \mathbb{P}$ which assigns to some of the clocks an integer constant or a parameter. For v a parameter valuation, we define a partial function $v(r) : \mathbb{X} \rightharpoonup \mathbb{Q}_+$ as follows: for each clock $x \in \mathbb{X}$, $v(r)(x) = k$ if $r(x) = k \in \mathbb{N}$ and $v(r)(x) = v(p) \in \mathbb{Q}_+$ if $r(x) = p$ a parameter. For a clock valuation w and a parameter valuation v, we denote by $[w]_{v(r)}$ the clock valuation obtained after applying $v(r)$.

III. Update-to-parameter Timed Automata

Timed automata [AD94] are an extension of finite-state automata augmented with clocks that can be compared to (usually) integer constants in guards (along edges), and that can be updated (usually) to 0 along edges. We extend this formalism by allowing clocks to be updated to *parameters*.

A. Syntax

Definition 1 (U2P-TA)**.** An *update-to-parameter timed automaton* (U2P-TA) $\mathcal{A}$ is a tuple $\mathcal{A} = (\Sigma, L, l_0, \mathbb{X}, \mathbb{P}, E)$, where:

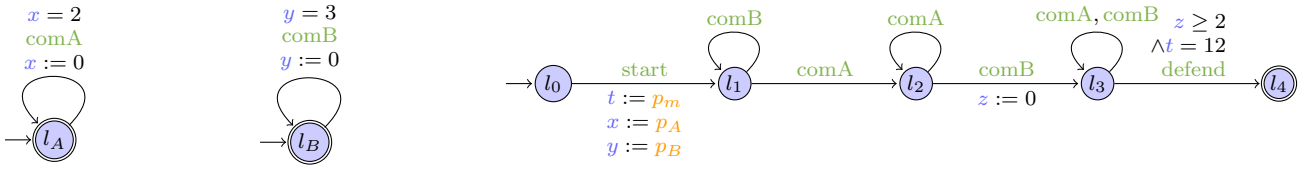
1) Σ is a finite set of actions,
2) L is a finite set of locations,
3) $l_0 \in L$ is the initial location,
4) $\mathbb{X}$ is a finite set of clocks,
5) $\mathbb{P}$ is a finite set of parameters,
6) E is a finite set of edges $e = (l, g, a, r, l')$ where $l, l' \in L$ are the source and target locations, g is a non-parametric guard, $a \in \Sigma$ and $r : \mathbb{X} \rightharpoonup \mathbb{N} \cup \mathbb{P}$ is a parametric update function.

In a concurrent setting, timed automata can be synchronized on shared actions. It is well-known that the product of several TAs gives a TA (see *e. g.,* [Mil00]). Moreover, real-time physical systems modeled with TAs can be implemented and timed properties checked using *e. g.,* Uppaal [BLL+95] or IMITATOR [AFKS12]. Similarly, our U2P-TAs can be synchronized the same way, and their product gives a U2P-TA. Their implementation is discussed in Section V.

Example 1. Consider the U2P-TA in Fig. 1c with five locations, four clocks (x, y, z and t) and three parameters (p_m, p_A, p_B). Observe that all three parameters are used in an update along the edge from l_0 and l_1.

As a motivating toy example, consider the case of a PhD student aiming at obtaining the authorization of her/his university in order to defend before December (assuming the system is starting at any moment). Two committees need to give their authorization sequentially (A then B), and the student must bring both authorizations to the administration two months ahead of the defense. Committee A (resp. B) meets periodically every two (resp. three) months, which is depicted in Figs. 1a and 1b, assuming time units are months.

The student workflow is modeled by the U2P-TA in Fig. 1c, synchronizing with both committees using actions comA and comB (clock x is shared between committee A and the student automaton, while y is shared between B and the student).



(a) Committee A (b) Committee B (c) A PhD student's defense workflow

Fig. 1: A motivating example of U2iP-TA

First, the student starts the process at time p_m, using the parametric update $t := p_m$. At the same time, we set the current clock of both committees to an unknown time; that is, assuming $p_A \in [0, 2]$ and $p_B \in [0, 3]$, the last occurrence of committee A (resp. B) is p_A (resp. p_B) or, put differently, the next occurrence of committee A is $2 - p_A$ (resp. $3 - p_B$). This allows us to analyze symbolically the system, by setting the clock t, that acts as a global timer, to the accurate student start date p_m, while assuming an unknown situation of the two periodic committees. Then, the student waits for the next commission A, and gets the authorization, moving to location l_2; then, (s)he waits for the next commission B, and gets the authorization, moving to location l_3. Finally, (s)he waits two more months (using $z \geq 2$) and defends in December (encoded by $t = 12$) in location l_4. The synchronization on comA and/or comB on self-loops allows the system to remain non-blocking.

The purpose of this analysis is to understand when in the year the student may start the workflow in order to be able to defend in December, depending on the current "offset" of the committees. That is, we want to synthesize the parameter valuations for p_m, p_A and p_B such that the system may eventually reach l_4.

Given a parameter valuation v, we denote by $v(\mathcal{A})$ the structure where all occurrences of a parameter p_i have been replaced by $v(p_i)$. If $v(\mathcal{A})$ is such that all constants in updates are integers, then $v(\mathcal{A})$ is an updatable timed automaton (see [BDFP04, Section 3.1]). In the following, we simply refer to an updatable timed automaton as a timed automaton. In the following, we consider a timed automaton any structure $v(\mathcal{A})$, by assuming a rescaling of the constants: by multiplying all constants in $v(\mathcal{A})$ by their least common denominator, we obtain an equivalent (integer-valued) timed automaton.

A *bounded* U2P-TA is a U2P-TA with a bounded parameter domain that assigns to each parameter a minimum integer bound and a maximum integer bound. That is, each parameter p_i ranges in an interval $[a_i, b_i]$, with $a_i, b_i \in \mathbb{N}$. Hence, a bounded parameter domain is a hyperrectangle of dimension M.

B. Semantics of timed automata

Let us now recall the concrete semantics of TAs.

Definition 2 (Concrete semantics of a TA)**.** Given a U2P-TA $\mathcal{A} = (\Sigma, L, l_0, \mathbb{X}, \mathbb{P}, E)$, and a parameter valuation v, the concrete semantics of $v(\mathcal{A})$ is given by the timed transition system $(S, s_0, \rightarrow)$, with

- $S = \{(l, w) \in L \times \mathbb{R}_+^H\}$, $s_0 = (l_0, \vec{0})$
- $\rightarrow$ consists of the discrete and (continuous) delay transition relations:
  - discrete transitions: $(l, w) \overset{e}{\mapsto} (l', w')$, if $(l, w), (l', w') \in S$, there exists $e = (l, g, a, r, l') \in E$, $w' = [w]_{v(r)}$, and $w \models g$.
  - delay transitions: $(l, w) \overset{d}{\mapsto} (l, w + d)$, with $d \in \mathbb{R}_+$, if $\forall d' \in [0, d], (l, w + d') \in S$.

Moreover we write $(l, w) \xrightarrow{e} (l', w')$ for the combination of a delay and a discrete transition where $((l, w), e, (l', w')) \in \rightarrow$ if $\exists d, w'' : (l, w) \overset{d}{\mapsto} (l, w'') \overset{e}{\mapsto} (l', w')$.

Given a TA $v(\mathcal{A})$ with concrete semantics $(S, s_0, \rightarrow)$, we refer to the states of S as the *concrete states* of $v(\mathcal{A})$. A (concrete) *run* of $v(\mathcal{A})$ is a possibly infinite alternating sequence of concrete states of $v(\mathcal{A})$ and edges starting from the initial concrete state s_0 of the form $s_0 \xrightarrow{e_0} s_1 \xrightarrow{e_1} \cdots \xrightarrow{e_{m-1}} s_m \xrightarrow{e_m} \cdots$, such that for all $i = 0, 1, \ldots$, $e_i \in E$, and $(s_i, e_i, s_{i+1}) \in \rightarrow$. Given a state $s = (l, w)$, we say that s is reachable (or that $v(\mathcal{A})$ reaches s) if s belongs to a run of $v(\mathcal{A})$. By extension, we say that l is reachable in $v(\mathcal{A})$, if there exists a state (l, w) that is reachable.

Throughout this paper, let K denote the largest constant in a given U2P-TA, *i. e.,* the maximum between the largest constant compared to a clock in a guard or used in an update, and the largest bound of a parameter (if the U2P-TA is bounded).

C. Problem

In this paper, we address the two following problems, given P a class of problems (*e. g.,* reachability, unavoidability, TCTL model-checking):

P-emptiness problem:
INPUT: a U2P-TA $\mathcal{A}$ and an instance ϕ of P
PROBLEM: is the set of valuations v such that $v(\mathcal{A})$ satisfies ϕ empty?

P-universality problem:
INPUT: a U2P-TA $\mathcal{A}$ and an instance ϕ of P
PROBLEM: are all valuations v such that $v(\mathcal{A})$ satisfies ϕ?

We mainly focus on reachability (EF) and unavoidability (AF) [JLR15]. EF-emptiness asks, given a U2P-TA $\mathcal{A}$ and a location l whether the set of valuations v such that there is a run in $v(\mathcal{A})$ reaching l is empty? It is equivalent to

AG-universality [And17]. More formally, the problem can be written as $\{v \mid \exists s_0 \xrightarrow{e_0} (l_1, w_1) \xrightarrow{e_1} \cdots \xrightarrow{e_{m-1}} (l, w)$ a run of $v(\mathcal{A})\} = \emptyset$?

AF-emptiness asks, given a U2P-TA $\mathcal{A}$ and a location l whether the set of valuations v such that all runs in $v(\mathcal{A})$ reach l is empty? It is equivalent to EG-universality [And17].

EF-universality asks, given a U2P-TA $\mathcal{A}$ and a location l whether all valuations v are such that there is a run in $v(\mathcal{A})$ reaching l? It is equivalent to AG-emptiness [And17].

Finally, AF-universality asks, given a U2P-TA $\mathcal{A}$ and a location l whether all valuations v are such that all runs in $v(\mathcal{A})$ reach l? It is equivalent to EG-emptiness [And17].

Beyond the theoretical decision problems above, an ultimate goal is the following computation problem.

P-synthesis problem:
INPUT: a U2P-TA $\mathcal{A}$ and an instance ϕ of P
PROBLEM: compute the set of valuations v such that $v(\mathcal{A})$ satisfies ϕ

Note that if EF-emptiness is undecidable, there is no hope for a useful and effective EF-synthesis procedure.

IV. Undecidability

In this section, we show that our extension of TAs with parametric updates leads to the undecidability of the EF-emptiness problem.

We show that any *bounded* (rational-valued) PTA can be transformed into a U2P-TA, and therefore that U2P-TAs are at least as expressive as (bounded) PTAs for which the EF-emptiness is known to be undecidable [Mil00].

Let us first recall PTAs [AHV93].

Definition 3 (PTA)**.** A *parametric timed automaton* (PTA) is a U2P-TA such that

1) every update function is a non-parametric update function;
2) guards along edges may be parametric guards.

Given a PTA $\mathcal{A}$ and a valuation v, we denote by $v(\mathcal{A})$ the structure where all occurrences of a parameter p_i have been replaced by $v(p_i)$. If $v(\mathcal{A})$ is such that all constants in guards are integers, then $v(\mathcal{A})$ is a *timed automaton*. Again, as for U2P-TA, given a PTA $\mathcal{A}$, we may denote as a timed automaton any structure $v(\mathcal{A})$, by assuming a rescaling of the constants. The semantics of PTA is identical to that of U2P-TA, since it is given in Definition 2 for a valuated PTA, *i. e.,* a timed automaton.

The main idea of our proof is as follows: suppose that, in a PTA, we want to measure a (parametric) duration p. Then we can update a clock x to 0 and then test it with a guard $x = p$. But provided we know an upper bound K on p, we could, with a U2P-TA, update clock x to $K - p$ and test it with a guard $x = K$ instead. Now, since we do not allow linear expressions in updates, we instead replace $K - p$ with a new parameter p' and prove that the existence of a valuation for p' in the U2P-TA such that the property holds, is equivalent to that of a valuation for p in the initial PTA. This idea extends to other comparison operators than $=$ and its practical development requires a few clock and parameter duplications.

Let $\mathcal{A} = (\Sigma, L, l_0, \mathbb{X}, \mathbb{P}, E)$ be a bounded PTA and K its largest constant. Let us define the following U2P-TA: $\mathcal{A}' = (\Sigma, L \cup \{l'_0\}, l'_0, \mathbb{X}', \mathbb{P}', E')$, which has the same actions as $\mathcal{A}$. For each $x \in \mathbb{X}$, $\mathbb{X}'$ contains x and a duplicate x_p for each parameter p to which x is compared in $\mathcal{A}$. $\mathbb{P}'$ contains all parameters in $\mathbb{P}$, as well as one extra parameter per clock in $\mathbb{X}$; given a clock $x \in \mathbb{X}$, we denote by p_x its corresponding extra parameter in $\mathbb{P}'$.

Let us now build E', initially containing all edges of E, and then modified as follows. Let x be a clock. Let $e = (l_1, g, a, r, l_2)$ be an edge of $\mathcal{A}$. If $r(x) = 0$, we perform the following modifications: first, we also update x_p to p_x along e, *i. e.,* $r(x_p) = p_x$. In addition, for any edge e' comparing clock x to parameter p in its guard, we replace $x \bowtie p$ with $x \bowtie K$. All other updates and non-parametric guards remain unchanged. Finally, we add one additional location l'_0 to the locations L of $\mathcal{A}$, which will be the new initial location, and one new additional edge from l'_0 to the former inital location l_0 of $\mathcal{A}$, with guard $x = 0$ for any clock $x \in \mathbb{X}$ and which updates for all clock $x \in \mathbb{X}$, x_p to p_x.

Example 2. An example of this construction is shown in Fig. 2, where we assume that p_1 is bounded in $[2, 5]$ and $p_2 \in [0, 12]$—therefore $K = 12$. For example, $12 - p_{1x}$ plays the role of p_1, and $12 - p_{2x}$ plays the role of p_2.

Since the initial sets of clocks $\mathbb{X}$ and $\mathbb{P}$ are finite and our set of linear constraints is finite, we only add a finite number of clocks and parameters to the new automaton. Finally, $\mathcal{A}_{RtP}$ is a U2P-TA. We denote by $\mathcal{A}_{RtP} = \mathsf{UtP}(\mathcal{A})$ this transformation.

Note that our transformation adds to the initial system in the worst case one parameter and one clock for each comparison to a parameter, *i. e.,* $|\mathbb{P}'| + |\mathbb{X}'| \leq |\mathbb{P}| + |\mathbb{X}| + 2 \times |\mathbb{P}| \times |\mathbb{X}|$.

In order to show that EF-emptiness is undecidable for U2P-TA, we prove the following behavior: a goal location is reached by a run in a U2P-TA $\mathcal{A}$, if and only if there is a run in $\mathsf{UtP}(\mathcal{A})$ reaching it.

Consider the automaton presented in Fig. 3a. Given a parameter valuation $v(p)$, we duplicate the clock x to x_p and update it to p_x where x is updated to 0. When x is compared to p, we replace this comparison by x_p compared to p_x, providing the automaton presented in Fig. 3b. During an execution of Fig. 3a accessing l_2, the time elapsed since the update of x until its comparison to p is $v(p)$. During an execution of Fig. 3b accessing l_2, the time elapsed since the update of x_p until its comparison to p_x is $K - v(p_x)$. We define the parameter valuation $v'(p) = K - v(p_x)$. With this construction, there is a parameter valuation v such that there is a run from l_0 to l_2 in Fig. 3a iff there is a parameter valuation v' as defined such that there is a run from l_0 to l_2 in Fig. 3b.

Proposition 1. *Let $\mathcal{A}$ be a bounded PTA, K its maximum constant, v be a parameter valuation, and $v' = K - v$. Let l be a goal location.*

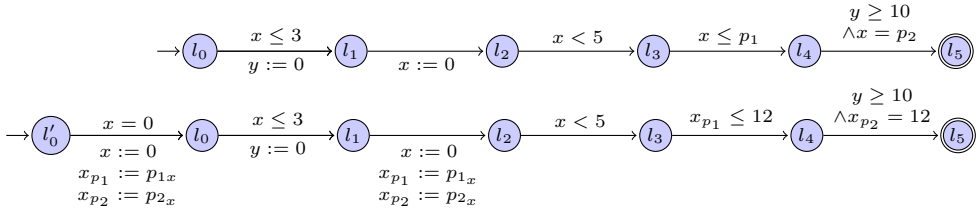


Fig. 2: A bounded PTA $\mathcal{A}$ (above) and its equivalent $\mathsf{UtP}(\mathcal{A})$ (below)

(a) A PTA $\mathcal{A}$

(b) An U2P-TA $\mathsf{UtP}(\mathcal{A})$

Fig. 3: A PTA $\mathcal{A}$ and its equivalent $\mathsf{UtP}(\mathcal{A})$

There is a run in $v(\mathcal{A})$ reaching l iff there is a run in $v'(\mathsf{UtP}(\mathcal{A}))$ reaching l.

Proof. Let ρ be a finite run of $v(\mathcal{A})$ ending in a concrete state (l, w) and let $\sigma = e_1 \dots e_n$ be the corresponding sequence of edges taken by ρ. We build by induction on n, a run ρ' in $v'(\mathsf{UtP}(\mathcal{A}))$ ending in a concrete state (l, w') such that for all $x \in \mathbb{X}, w'(x) = w(x)$ and for all clock $x' \in \mathbb{X}' \setminus \mathbb{X}, w'(x_p) = K - v(p) + w(x)$.

If $n = 0$, then ρ' consists only of the additional initial edge of $\mathsf{UtP}(\mathcal{A})$, which clearly sets all clocks to the adequate values.

Suppose now that we have built ρ' for size n and consider a run ρ with $n+1$ edges. Then ρ consists of a run ρ_1, ending in (l_1, w_1) with n edges followed by a delay d and finally a discrete transition along the last edge e. From the induction hypothesis, we can build an equivalent run ρ'_1 in $\mathsf{UtP}(\mathcal{A})$ ending in (l_1, w'_1), such that for all $x \in \mathbb{X}, w'_1(x) = w_1(x)$ and for all clock $x_p \in \mathbb{X}' \setminus \mathbb{X}, w'_1(x_p) = K - v(p) + w_1(x)$. Let w_2 (resp. w'_2) be the clock valuation obtained in $\mathcal{A}$ (resp. $\mathsf{UtP}(\mathcal{A})$) after the delay d. By construction, the part of the guard of e comparing clocks in $\mathbb{X}$ to constants is satisfied by w'_2 since it is the same as in $\mathcal{A}$. Further, for each clock $x \in \mathbb{X}$, such that $x \bowtie p$ along e in $\mathcal{A}$, we have instead $x_p \bowtie K$ along the modified e in $\mathsf{UtP}(\mathcal{A})$. But $w'_2(x_p) = K - v(p) + w_2(x)$, so the latter comparison is equivalent to $K - v(p) + w_2(x) \bowtie K$, *i. e.,* $w_2(x) \bowtie v(p)$. So, since the guard is satisfied in $\mathcal{A}$ by w_2, the corresponding guard is satisfied in $\mathsf{UtP}(\mathcal{A})$ by w'_2. Then clocks in $\mathbb{X}$ are updated normally, and for all clocks $x_p \in \mathbb{X}' \setminus \mathbb{X}$, we have an update to $v'(p_x) = K - v(p)$, which concludes the induction.

The other direction, starting from a run in $\mathsf{UtP}(\mathcal{A})$, is similar. □

Theorem 1. *The* EF*-emptiness problem is undecidable for bounded U2P-TAs.*

Proof. From the undecidability of EF-emptiness for bounded PTAs [Mil00]. □

We now show that this result can be extended to the full class of (unbounded) U2P-TAs.

Theorem 2. *The* EF*-emptiness problem is undecidable for U2P-TAs.*

Proof. Similarly to the proof of [ALR16b, Proposition 8], we claim that a bounded U2P-TA can be easily simulated using an unbounded U2P-TA. We present a gadget in Fig. 4 that uses two clocks (that can be clocks used by the PTA) and two transitions that can be added before the initial location of any unbounded U2P-TA, and ensures a parameter p is bounded, *i. e.,* given two integer constants min and max we have $p \in [min, max]$. We need one gadget per parameter; these gadgets can be branched sequentially before the initial location of an unbounded U2P-TA, and all clocks must be updated to 0 before entering the initial location.

The gadget works as follows: when taking the first transition from l_0 to l_1, clock x is updated to p and clock y to 0. The transition from l_1 to l_2 can be taken if and only if in a 0-delay ensured by the guard $y = 0$, we have that $x \leq max$ and $min \leq x$. This means there is a run from l_0 to l_2 if and only if there is a parameter valuation v such that $min \leq v(p) \leq max$, which in other words means that the parametric domain is bounded.

As from Theorem 1 the EF-emptiness problem is undecidable for bounded U2P-TA, and as any bounded U2P-TA can be expressed using a U2P-TA, we conclude that the EF-emptiness problem is undecidable for unbounded U2P-TA.

□

Corollary 1. *The* AF*-emptiness problem is undecidable for U2P-TAs.*

Proof. The AF-emptiness problem is undecidable for PTAs as it is proven undecidable for one of its subclasses in [JLR15]. Since we can encode a PTA into a U2P-TA, it is undecidable for the former. □

Corollary 2. AF, EF*-universality problems are undecidable for U2P-TAs.*

Proof. In [ALR16a], EG, AG-emptiness problems are proven undecidable for PTAs. As AF, EF-universality are their equiv-

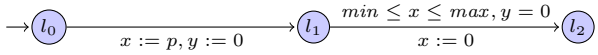


Fig. 4: A gadget that ensures a parameter p is bounded by min and max

alent respectively, they are also undecidable for PTAs, and therefore for U2P-TAs. □

V. Decidability

Let us now show that, when parameters are restricted to (unbounded) integers, the EF-emptiness problem becomes PSPACE-complete.

If parameters in an U2P-TA only have (possibly unbounded) integer valuations, we say it is an U2iP-TA. Note that once valuated by an integer parameter valuation v, an U2iP-TA is an updatable timed automaton with updates to integer constants, as defined in [BDFP04, Section 3.1]. Hence *clock regions* are still topical in this context [BDFP04, Section 5.1]. Let us recall the notion of clock region [AD94]. Given a clock x and a clock valuation w, recall that $\lfloor w(x) \rfloor$ denotes the integer part of $w(x)$ while $frac(w(x))$ denotes its fractional part.

Definition 4 (clock region)**.** For two clock valuations w and w', $\sim$ is an equivalence relation defined by: $w \sim w'$ iff

1) for all clock x, either $\lfloor w(x) \rfloor = \lfloor w'(x) \rfloor$ or $w(x), w'(x) > K$;
2) for all clocks x, y with $w(x), w(y) \leq K$, $frac(w(x)) \leq frac(w(y))$ iff $frac(w'(x)) \leq frac(w'(y))$;
3) for all clock x with $w(x) \leq K$, $frac(w(x)) = 0$ iff $frac(w'(x)) = 0$.

A *clock region* R_c is an equivalence class of $\sim$.

Two clock valuations in the same clock region reach the same region by time elapsing, satisfy the same guards and thus can take the same transitions [AD94].

Theorem 3. *The set of parameter valuations for which a given location is reachable is effectively computable for U2iP-TA.*

Proof. We first need an intermediate lemma:

Lemma 1. *Let $\mathcal{A}$ be an U2iP-TA. Let K be the greatest constant in $\mathcal{A}$. Let l be a goal location. Let v, v' be two rational parameter valuations s.t. for all parameter p, either $v(p) = v'(p)$ or $v(p) > K$ and $v'(p) > K$. There is a run in $v(\mathcal{A})$ reaching (l, w) iff there is a run in $v'(\mathcal{A})$ reaching (l, w') s.t. at each state, two clock valuations of ρ and ρ' are in the same clock region and location.*

Proof. By induction on the length of the run. Let v, v' be such parameter valuations.

For a run of length 0 of $v(\mathcal{A})$, there is a run of length 0 of $v'(\mathcal{A})$ reaching the initial location. If there is a run of length 0 of $v'(\mathcal{A})$, there is a run of length 0 of $v(\mathcal{A})$ reaching the initial location.

Now, suppose the result holds for every run of length i. Assume a run of $v(\mathcal{A})$ of length $i+1$, with a prefix ρ of length i reaching (l_i, w_i) followed by a state obtained using edge $e = (l_i, g, a, r, l_{i+1})$. That is, the run is of the form $\rho \xrightarrow{e} (l_{i+1}, w_{i+1})$.

By induction hypothesis, let ρ' be a run of $v'(\mathcal{A})$ reaching (l_i, w'_i) s.t. at each state, two clock valuations of ρ and ρ' are in the same clock region and location.

Now if for all clock x, no $w_i(x)$ is the result of a parametric update, then trivially $w_i \models g$ and as $w_i \sim w'_i$, $w'_i \models g$. Alternatively, suppose for some x and parameter p, we have $w_i(x) = v(p)$. If $v(p) < K+1$ and $w_i \models g$, since $v'(p) = v(p)$ then as $w_i \sim w'_i$, $w'_i \models g$. If $v(p) \geq K+1$ and $w_i \models g$, since $v'(p) \geq K+1$ then as $w_i \sim w'_i$, $w'_i \models g$. We treat the case of multiple updates of clocks to parameters in e the same way. Finally, we can take the transition e with the same delay. Hence $\xrightarrow{e} (l_{i+1}, w'_{i+1})$ is a run of $v'(\mathcal{A})$ of length $i+1$ reaching l_{i+1} with the same actions, locations, delays and at each state, two clock valuations of ρ and ρ' are in the same clock region and location.

The other way is a direct consequence of the previous paragraph and the definition of the clock regions.

□

We can now go back to the proof of Theorem 3. Let $\mathcal{A}$ be an U2iP-TA and K be the greatest constant in $\mathcal{A}$. Now let v be a (integer) parameter valuation. Since $v(\mathcal{A})$ is an updatable timed automaton, the reachability of a given state (l, w) is decidable [BDFP04, Section 5]. It is sufficient to enumerate all integer valuations s.t. for each parameter p, $v(p) \leq K+1$. Indeed, from Lemma 1 a parameter valuation v with $v(p) > K+1$ allows to take the same transitions and reach the same guards as the parameter valuation v' s.t. for all $p' \neq p$, $v(p') = v'(p')$ and $v'(p) = K+1$ so we can replace such parameter valuations by a valuation v' as defined previously. In conclusion, there is a finite number of parameter valuations to test to obtain the full set of valuations for which the goal location is reachable. □

Proposition 2. *The EF-emptiness problem is PSPACE-complete for U2iP-TAs.*

Proof. Since we can synthesize exactly the set of parameter valuations for which the goal location is reachable using Theorem 3, the decidability of the EF-emptiness follows immediately.

Let us now have a look at the complexity of the EF-emptiness problem for U2iP-TA. First, since a TA is a special case of U2iP-TA with no parametric update, we have the PSPACE-hardness for EF-emptiness in our U2iP-TA [AD94]. Now, let G be a set of goal locations of $\mathcal{A}$. Consider the non-deterministic Turing machine that:

1) takes $\mathcal{A}$, G and K as input
2) non-deterministically "guesses" an integer valuation v bounded by $K+1$ and writes it to the tape

3) overwrite on the tape each parameter p by $v(p)$, giving the updatable TA $v(\mathcal{A})$
4) solves reachability in $v(\mathcal{A})$ for G
5) accepts iff the result of the previous step is "yes".

The machine accepts iff there is an integer valuation v bounded by $K+1$ and a run in $v(\mathcal{A})$ reaching a location $l \in G$.

The size of the input is $|\mathcal{A}| + |G| + |K|$, using $|.|$ to denote the size in bits of the different objects. There are at most $(K+1)^M$ possible valuations, where M is the number of parameters in $\mathcal{A}$. Storing the valuation at step 2 uses at most $M \times |K+1|$ additional bits, which is polynomial w.r.t. the size of the input. Step 4 also needs polynomial space from [BDFP04]. So globally this non-deterministic machine runs in polynomial space. Finally, by Savitch's theorem we have PSPACE = NPSPACE [Sav70], and the expected result. □

The following result is direct from Theorem 3:

Corollary 3. *The* EF*-universality problem is decidable for U2iP-TAs.*

Proof. Using Lemma 1 (see proof of Theorem 3) given an U2iP-TA $\mathcal{A}$ and its greatest constant in $\mathcal{A}$, there is a finite number of parameter valuations to test. Therefore given a goal location l, it is sufficient to test whether for all parameter valuations, there is a run reaching l in the valuated instance of $\mathcal{A}$. □

We state also the two following corollaries that fulfill the last unknown decision problems considered in this paper for U2P-TAs.

Corollary 4. *The set of parameter valuations for which a given location is unavoidable is effectively computable for U2iP-TA.*

Proof. Let $\mathcal{A}$ be an U2iP-TA and v a parameter valuation. As we use in our construction the same clock regions as in [AD94], suppose there is a run in $v(\mathcal{A})$ reaching a location l, then all runs going through the same clock regions are equivalent—they satisfy the same guards, and end in the same region after an update and after letting time elapse. Moreover, using the construction of the region automaton of [AD94], it is sufficient to test whether all runs in the region automaton of $\mathcal{A}$ reach l, which are in finite number. Using the same reasoning as in the proof of Theorem 3 we obtain our result. □

Corollary 4 leads to the decidability of the AF-emptiness problem. Following the same reasoning as in Theorem 3, we state the last but not least result of this paper:

Corollary 5. *The* AF*-emptiness and* AF*-universality problems are decidable for U2iP-TAs.*

Proof. Given an U2iP-TA $\mathcal{A}$ and using the same reasoning as in the previous proof and the region automaton of [AD94], we can test whether all runs in this region automaton reach l, which are in finite number. As there is a finite number of parameter valuations to test, we can compute the set of parameter valuations such that all runs reach l (*i. e.,*, AF-synthesis) from Corollary 4. Testing the emptiness of the obtained set of parameter valuations gives AF-emptiness. Given a goal location l, it is sufficient to test whether for all parameter valuations, there is a run reaching l in the valuated instance of $\mathcal{A}$ to decide AF-universality. □

Implementation in IMITATOR

U2P-TAs (and naturally U2iP-TA) are supported by IMITATOR [AFKS12], a parametric model checker taking as input extensions of parametric timed automata.

Passing Example 1 as input and using the reachability synthesis algorithm, IMITATOR synthesizes the following constraint:

$$p_B + 4 \geq p_m \wedge p_B \geq p_A + 1 \wedge p_B \leq 3$$
$$\vee$$
$$p_m \leq p_B + 7 \wedge p_A \leq 2 \wedge p_B \leq p_A + 1$$

The first conjunction of inequalities states that, if the committee B is the next to meet (which is encoded by $p_B \geq p_A + 1$, and could also be written as $3 - p_B \leq 2 - p_A$), then the month p_m at which the student starts the process should be less than 4 plus the number of months since the last occurrence of committee B. (The last inequality simply recalls that p_B is less than or equal to 3). The second conjunction of inequalities states that, if the committee A is the next to meet, then the month p_m at which the student starts the process should be less than 7 plus the number of months since the last occurrence of committee B.

For any such valuation, there exists a run of the system (*i. e.,* a configuration of the committees dates respecting their respective periods) such that the student may defend in December. Also note that, if we add proper invariants[1], then the system becomes completely deterministic and the valuations for which there exists a run reaching l_4 are also such that all runs reach l_4 (since there exists only one run), and therefore the student is *guaranteed* to be able to defend in December for any of these valuations.

We can also study a situation where the system is only partially parameterized: assume $p_m = 6$, *i. e.,* the student will start the process in June in any case. The constraint encoding the current state of committees A and B is given by:

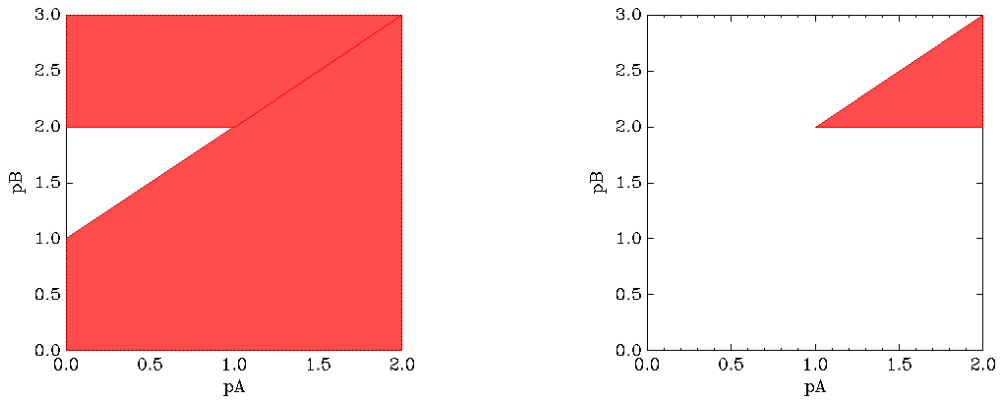
$$p_A \leq 2 \wedge p_B \leq p_A + 1$$
$$\vee$$
$$p_B \geq 2 \wedge p_B \leq 3 \wedge p_B \geq p_A + 1$$

A graphical visualization (output by IMITATOR) is given in Fig. 5a (plain red depicts good valuations, *i. e.,* for which the student may defend in December).

Alternatively, if $p_m = 9$ (*i. e.,* the student starts the process in September), then the constraint on p_A and p_B is as follows:

$$p_B \geq 2 \wedge p_A \leq 2 \wedge p_A + 1 \geq p_B$$

[1]Precisely, $x \leq 2$ in committee A, $y \leq 3$ in committee B, and $t \leq 12$ in the student automaton.



(a) $p_m = 6$ (b) $p_m = 9$

Fig. 5: Graphical visualization in two dimensions of the parameter synthesis of Example 1

A graphical visualization is given in Fig. 5b.

Finally note that this entire example is not restricted to integer-valued parameters (rational-valued months can be used to denote finer time grain, *e. g.,* days or even hours), and it therefore falls in the undecidable case of Theorem 1. Nevertheless, IMITATOR terminates here with an exact (sound and complete) result.

VI. Conclusion

In this paper we defined two new formalisms to model concurrent timed systems with uncertainty: U2P-TA for which we proved that the EF-emptiness problem is undecidable, even for bounded parameters, and U2iP-TA for which we proved that the EF-emptiness problem is PSPACE-complete. This discrepancy between integer-valued and rational-valued was already spotted in parametric timed automata: the EF-emptiness is decidable for integer-valued parameters with 1 parametric clock (*i. e.,* a clock compared to a parameter in at least one guard) and 3 non-parametric clocks [AHV93], while it becomes undecidable over rational-valued parameters [Mil00]. Similarly, the discrepancy between (rational-valued) bounded parameters and unbounded parameters is reminiscent of the recent result we showed for EG-emptiness ("is the set of valuations for which at least one maximal run remains in a given set of locations empty?"): this problem is decidable for bounded L/U-PTAs (a parameter is either used as an upper bound or a lower bound in guards) with rational-valued parameters, while it becomes undecidable for the full class of L/U-PTAs [AL17]. Furthermore, we extended our undecidability results to the EF-universality, AF-emptiness and AF-universality problems for U2P-TA, but also our decidability results to these same problems for U2iP-TA. This paper therefore handles a wide range of decision problems for U2P-TA. We assume that the decidability could be extended to the full TCTL model checking following a similar reasoning.

The fact that we allow update to parameters in the (possibly parametric) timed extensions of finite-state automata is quite new and, to the best of our knowledge, has not been investigated until now. Despite having an undecidability result when the parameter domain is rational, we believe this new formalism, improved with parameters allowed in guards, could become decidable even over rational-parameters if we add a few semantic restrictions. Indeed, reset-PTAs have been studied in [ALR16a] and are a promising subclass of PTA to extend. For this purpose, we would like to explore PTAs in which update to parameters is also allowed, and under which conditions the EF-emptiness problem could become decidable. Moreover, the semantic restrictions of reset-PTAs (a clock is updated to 0 whenever it is compared to a parameter) is in a way reminiscent to *initialized* rectangular hybrid automata (a variable is updated whenever its dynamic changes) presented in [HKPV98] and it would be interesting to study these systems in which we involve parameters. Therefore, extending our result to hybrid automata is also an interesting perspective.

Finally, beyond the toy aspect of Example 1, we believe that U2iP-TAs can be used to model scheduling problems for real-time systems subject to uncertainty, notably in the tasks offsets, as this is where we used parameters in Fig. 1.

References

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, April 1994.

[AFKS12] Étienne André, Laurent Fribourg, Ulrich Kühne, and Romain Soulat. IMITATOR 2.5: A tool for analyzing robustness in scheduling problems. In Dimitra Giannakopoulou and Dominique Méry, editors, *FM*, volume 7436 of *Lecture Notes in Computer Science*, pages 33–36. Springer, 2012.

[AHV93] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. Parametric real-time reasoning. In S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal, editors, *STOC*, pages 592–601, New York, NY, USA, 1993. ACM.

[AL17] Étienne André and Didier Lime. Liveness in L/U-parametric timed automata. In Alex Legay and Klaus Schneider, editors, *ACSD*, pages 9–18. IEEE, 2017.

[ALR16a] Étienne André, Didier Lime, and Olivier H. Roux. Decision problems for parametric timed automata. In Kazuhiro Ogata, Mark Lawford, and Shaoying Liu, editors, *ICFEM*, volume 10009 of *Lecture Notes in Computer Science*, pages 400–416. Springer, 2016.

[ALR16b] Étienne André, Didier Lime, and Olivier H. Roux. On the expressiveness of parametric timed automata. In *FORMATS*, volume 9984 of *Lecture Notes in Computer Science*, pages 19–34. Springer, 2016.

[And17] Étienne André. What's decidable about parametric timed automata? *International Journal on Software Tools for Technology Transfer*, 2017. To appear.

[BBLS15] Nikola Beneš, Peter Bezděk, Kim Gulstrand Larsen, and Jiří Srba. Language emptiness of continuous-time parametric timed automata. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *ICALP, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 69–81. Springer, July 2015.

[BDFP04] Patricia Bouyer, Catherine Dufourd, Emmanuel Fleury, and Antoine Petit. Updatable timed automata. *Theoretical Computer Science*, 321(2-3):291–345, August 2004.

[BL09] Laura Bozzelli and Salvatore La Torre. Decision problems for lower/upper bound parametric timed automata. *Formal Methods in System Design*, 35(2):121–151, 2009.

[BLL+95] Johan Bengtsson, Kim G. Larsen, Fredrik Larsson, Paul Pettersson, and Wang Yi. UPPAAL — a Tool Suite for Automatic Verification of Real–Time Systems. In *Proc. of Workshop on Verification and Control of Hybrid Systems III*, number 1066 in Lecture Notes in Computer Science, pages 232–243. Springer–Verlag, October 1995.

[BY03] Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In Jörg Desel, Wolfgang Reisig, and Grzegorz Rozenberg, editors, *Lectures on Concurrency and Petri Nets, Advances in Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*, pages 87–124. Springer, 2003.

[CL00] Franck Cassez and Kim Guldstrand Larsen. The impressive power of stopwatches. In Catuscia Palamidessi, editor, *CONCUR*, volume 1877 of *Lecture Notes in Computer Science*, pages 138–152. Springer, 2000.

[Doy07] Laurent Doyen. Robust parametric reachability for timed automata. *Information Processing Letters*, 102(5):208–213, 2007.

[HKPV98] Thomas A. Henzinger, Peter W. Kopke, Anuj Puri, and Pravin Varaiya. What's decidable about hybrid automata? *Journal of Computer and System Sciences*, 57(1):94–124, 1998.

[HRSV02] Thomas Hune, Judi Romijn, Mariëlle Stoelinga, and Frits W. Vaandrager. Linear parametric model checking of timed automata. *Journal of Logic and Algebraic Programming*, 52-53:183–220, 2002.

[JLR15] Aleksandra Jovanović, Didier Lime, and Olivier H. Roux. Integer parameter synthesis for timed automata. *IEEE Transactions on Software Engineering*, 41(5):445–461, 2015.

[Mil00] Joseph S. Miller. Decidability and complexity results for timed automata and semi-linear hybrid automata. In Nancy A. Lynch and Bruce H. Krogh, editors, *HSCC*, volume 1790 of *Lecture Notes in Computer Science*, pages 296–309. Springer, 2000.

[Sav70] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970.