Decision Problems for Parametric Timed Automata

Étienne André, Didier Lime, Olivier Henri Roux

Decision Problems for Parametric Timed Automata*

Étienne André[1,2], Didier Lime[1], and Olivier H. Roux[1]

[1] École Centrale de Nantes, IRCCyN, CNRS, UMR 6597, France
[2] Université Paris 13, Sorbonne Paris Cité, LIPN, CNRS, UMR 7030, F-93430, Villetaneuse, France

Abstract. Parametric timed automata (PTAs) allow to reason on systems featuring concurrency and timing constraints making use of parameters. Most problems are undecidable for PTAs, including the parametric reachability emptiness problem, *i. e.,* whether at least one parameter valuation allows to reach some discrete state. In this paper, we first exhibit a subclass of PTAs (namely integer-points PTAs) with bounded rational-valued parameters for which the parametric reachability emptiness problem is decidable. Second, we present further results improving the boundary between decidability and undecidability for PTAs and their subclasses.

1 Introduction

Timed automata (TAs) [AD94] are a powerful formalism that extend finite-state automata with clocks (real-valued variables evolving linearly) that can be compared with integer constants in locations ("invariants") and along transitions ("guards"); additionally, some clocks can be reset to 0 along transitions. Many interesting problems for TAs (including the reachability of a location) are decidable. However, the classical definition of TAs is not tailored to verify systems only partially specified, especially when the value of some timing constants is not yet known.

Parametric timed automata (PTAs) [AHV93] leverage this problem by allowing the specification and the verification of systems where some of the timing constants are parametric. PTAs extend TAs by allowing the use of integer- or rational-valued parameters in place of timing constants in guards and invariants. PTAs were used to model and verify a variety of case studies, from hardware circuits to communication protocols (see [And15] for a survey). This expressive power comes at the price of the undecidability of most interesting problems. The EF-emptiness problem ("does there exist a parameter valuation such that a given location is reachable?") is undecidable in general [AHV93], even when parameters are bounded [Mil00], even when only strict inequalities are used [Doy07], and

* This work is partially supported by the ANR national research program "PACS" (ANR-14-CE28-0002).

with a single integer-valued parameter [BBLS15]. However, bounding the number of parametric clocks and of parameters may yield decidability. The smallest known numbers of parametric clocks (*i. e.,* clocks compared with parameters), non-parametric clocks and parameters leading to undecidability are: three parametric clocks and one integer-valued parameter [BBLS15] or three parametric clocks and only one rational-valued parameter [Mil00], or only one parametric clock, three non-parametric clocks and one rational-valued parameter [Mil00].

In [HRSV02], L/U-PTAs are introduced as a subclass of PTAs where each parameter is either always compared to a clock as a lower bound in guards and invariants, or always as an upper bound. The EF-emptiness problem is decidable for L/U-PTAs. In [BL09], further results are proved for L/U-PTAs with integer-valued parameters: emptiness, finiteness and universality of the set of parameter valuations for which there exists an infinite accepting run are decidable. The AF-emptiness problem ("does there exist a parameter valuation for which a given location is eventually reached for any run?") is undecidable for L/U-PTAs [JLR15]. It is also shown in [JLR15] that the synthesis of the parameters reaching a given location in an L/U-PTA is intractable in practice. Two further subclasses have been defined in [BL09]: L-PTAs and U-PTAs, where all parameters are always lower bounds and upper bounds respectively.

In [JLR15], PTAs with bounded integer-valued parameters are considered. The problem of finding parameter valuations such that a given location is reachable or unavoidable becomes decidable, and two algorithms are provided that compute the exact such sets of integer valuations in a symbolic manner, *i. e.,* without performing an exhaustive enumeration. In [ALR15], it is shown that computing a parametric extrapolation of the integer hull of symbolic states allows one to synthesize (rational-valued) parameter valuations for bounded PTAs, guaranteeing the synthesis of at least all integer-valued valuations, but also sometimes most or even all rational-valued valuations.

Contribution L/U-PTAs is the only non-trivial[3] subclass of PTAs for which the EF-emptiness problem is decidable for an arbitrary number of clocks and parameters. However, other results are disappointing: undecidability of AF-emptiness, intractability of the synthesis [JLR15]. It is hence important to look for further subclasses of PTAs for which problems may be decidable. It is shown in [JLR15,ALR15] that integer points play a key role in decidability. Hence, our first contribution here is to investigate integer-points PTAs (IP-PTAs), that are PTAs where each symbolic state contains at least one integer point (*i. e.,* an integer valuation of the clocks and the parameters). Our intuition is successful: we prove that the EF-emptiness problem is decidable for bounded IP-PTAs (*i. e.,* with a bounded parameter domain), even when parameters are rational-valued. Although we show that it cannot be decided whether a bounded PTA is a (bounded) IP-PTA, we give two sufficient syntactic conditions: we show that bounded L/U-PTAs with non-strict inequalities are IP-PTAs and, more inter-

[3] The bounded integer PTAs of [JLR15] are arguably a trivial such subclass (even though the associated analysis techniques are not).

estingly, we introduce a new subclass of "reset-PTAs", that are also IP-PTAs, and for which, when bounded, the EF-emptiness problem is hence decidable too. This class is only the second syntactic subclass of PTAs (after L/U-PTAs) for which this problem is decidable.

Our second main contribution is to study several open problems for PTAs and several known subclasses (as well as the new class of IP-PTAs): we study here the emptiness and universality of reachability (EF), as well as unavoidability emptiness (AF). Emptiness is of utmost importance as, without decidability of the emptiness, exact synthesis is practically ruled out. Universality checks whether all parameter valuations satisfy a property, which is important for applications where the designer has no power on some valuations; this is the case of networks, where some latencies (*e. g.,* the transmission time of some packets) may be totally arbitrary. Among our results, we prove in particular that AF-emptiness is undecidable for both bounded IP-PTAs and bounded L/U-PTAs. Overall, we significantly enhance the knowledge we have of decidability problems for PTAs and subclasses.

Outline We first recall the necessary definitions in Section 2. Then, we introduce in Section 3 a new proof for the undecidability of the EF-emptiness problem for PTAs with a single rational-valued parameter; whereas this result is not essentially new (it has been known since [Mil00]), our original proof will be used in several other results of this paper. In addition, we extend this result (using a variant of our proof) to bounded PTAs with only non-strict inequalities which, to the best of our knowledge, is an original result. Then, we introduce the new class of IP-PTAs in Section 4, and study its properties. Finally, in part by using this new class, we prove in Section 5 several open results for L/U-PTAs and PTAs. We conclude in Section 6.

2 Preliminaries

2.1 Clocks, Parameters and Constraints

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

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

In the following, we assume $\prec \in \{<, \leq\}$ and $\bowtie \in \{<, \leq, \geq, >\}$. *lt* denotes a linear term over $X \cup P$ of the form $\sum_{1 \leq i \leq H} \alpha_i x_i + \sum_{1 \leq j \leq M} \beta_j p_j + d$, with

$x_i \in X$, $p_j \in P$, and $\alpha_i, \beta_j, d \in \mathbb{Z}$. *plt* denotes a parametric linear term over P, that is a linear term without clocks ($\alpha_i = 0$ for all i). A *constraint* C over $X \cup P$ is a conjunction of inequalities of the form $lt \bowtie 0$ (*i. e.,* a convex polyhedron). Given a parameter valuation v, $v(C)$ denotes the constraint over X obtained by replacing each parameter p in C with $v(p)$. Likewise, given a clock valuation w, $w(v(C))$ denotes the expression obtained by replacing each clock x in $v(C)$ with $w(x)$. We say that v *satisfies* C, denoted by $v \models C$, if the set of clock valuations satisfying $v(C)$ is nonempty. Given a parameter valuation v and a clock valuation w, we denote by $w|v$ the valuation over $X \cup P$ such that for all clocks x, $w|v(x) = w(x)$ and for all parameters p, $w|v(p) = v(p)$. We use the notation $w|v \models C$ to indicate that $w(v(C))$ evaluates to true. We say that C is *satisfiable* if $\exists w, v$ s.t. $w|v \models C$. An *integer point* is $w|v$, where w is an integer clock valuation, and v is an integer parameter valuation. We define the *time elapsing* of C, denoted by $C^{\nearrow}$, as the constraint over X and P obtained from C by delaying all clocks by an arbitrary amount of time. Given $R \subseteq X$, we define the *reset* of C, denoted by $[C]_R$, as the constraint obtained from C by replacing with 0 the value of the clocks in R, and keeping the value of other clocks unchanged. We denote by $C{\downarrow_P}$ the projection of C onto P, *i. e.,* obtained by eliminating the clock variables (*e. g.,* using the Fourier-Motzkin algorithm [**?**]).

A *guard* g is a constraint over $X \cup P$ defined by inequalities of the form $x \bowtie z$, where z is either a parameter or a constant in $\mathbb{Z}$.

A *zone* is a polyhedron over a set of variables V (usually clocks) in which all constraints on variables are of the form $x \bowtie k$ (rectangular constraints) or $x_i - x_j \bowtie k$ (diagonal constraints), where $x_i \in V$, $x_j \in V$ and k is an integer. Operations on zones are well-documented (see *e. g.,* [BY04]).

A *parametric zone* is a convex polyhedron over $X \cup P$ in which all constraints on variables are of the form $x \bowtie plt$ (parametric rectangular constraints) or $x_i - x_j \bowtie plt$ (parametric diagonal constraints), where $x_i \in X$, $x_j \in X$ and *plt* is a parametric linear term over P.

2.2 Parametric Timed Automata

Syntax

Definition 1. *A PTA $\mathcal{A}$ is a tuple $\mathcal{A} = (\Sigma, L, l_0, X, P, I, E)$, where: i) Σ is a finite set of actions, ii) L is a finite set of locations, iii) $l_0 \in L$ is the initial location, iv) X is a finite set of clocks, v) P is a finite set of parameters, vi) I is the invariant, assigning to every $l \in L$ a guard $I(l)$, vii) E is a finite set of edges $e = (l, g, a, R, l')$ where $l, l' \in L$ are the source and target locations, $a \in \Sigma$, $R \subseteq X$ is a set of clocks to be reset, and g is a guard.*

We say that a PTA is *closed* if all its guards and invariants use only non-strict constraints. Note that the grammar of constraints does not include negation so this restriction is meaningful, and that $=$ defines closed constraints.

Given a parameter valuation v, we denote by $v(\mathcal{A})$ the non-parametric timed automaton where all occurrences of a parameter p_i have been replaced by $v(p_i)$.

Concrete Semantics

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

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

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

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 an alternating sequence of concrete states of $v(\mathcal{A})$ and edges starting from the initial concrete state s_0 of the form $s_0 \overset{e_0}{\mapsto} s_1 \overset{e_1}{\mapsto} \cdots \overset{e_{m-1}}{\mapsto} s_m$, such that for all $i = 0, \ldots, m-1$, $e_i \in E$, and $(s_i, e_i, s_{i+1}) \in \mapsto$. Given a concrete state $s = (l, w)$, we say that s is reachable (or that $v(\mathcal{A})$ reaches s) if s belongs to a concrete run of $v(\mathcal{A})$. By extension, we say that l is reachable in $v(\mathcal{A})$.

Symbolic semantics Let us now recall the symbolic semantics of PTAs (see *e. g.,* [ACEF09]). A symbolic state is a pair (l, C) where $l \in L$ is a location, and C its associated parametric zone. The initial symbolic state of $\mathcal{A}$ is $\mathbf{s}_0 = (l_0, (\bigwedge_{1 \leq i \leq H} x_i = 0)^{\nearrow} \wedge I(l_0))$.

The symbolic semantics relies on the Succ operation. Given a symbolic state $\mathbf{s} = (l, C)$ and an edge $e = (l, g, a, R, l')$, $\mathsf{Succ}(\mathbf{s}, e) = (l', C')$, with $C' = \left([(C \wedge g)]_R \wedge I(l')\right)^{\nearrow} \wedge I(l')$..

A symbolic run of a PTA is an alternating sequence of symbolic states and edges starting from the initial symbolic state, of the form $\mathbf{s}_0 \overset{e_0}{\Rightarrow} \mathbf{s}_1 \overset{e_1}{\Rightarrow} \cdots \overset{e_{m-1}}{\Rightarrow} \mathbf{s}_m$, such that for all $i = 0, \ldots, m-1$, $e_i \in E$, and $\mathbf{s}_{i+1}$ belongs to $\mathsf{Succ}(\mathbf{s}_i, e)$. Given a symbolic state $\mathbf{s}$, we say that $\mathbf{s}$ is reachable if $\mathbf{s}$ belongs to a symbolic run of $\mathcal{A}$. In the following, we simply refer to symbolic states belonging to a run of $\mathcal{A}$ as symbolic states of $\mathcal{A}$.

2.3 Subclasses of PTAs

In this paper, we will sometimes consider *bounded* PTAs, *i. e.,* PTAs 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.

Let us now recall L/U-PTAs [HRSV02,BL09].

Definition 3 (L/U-PTA [HRSV02]). *An L/U-PTA is a PTA where the set of parameters is partitioned into lower-bound parameters and upper-bound parameters. A lower- (resp. upper-)bound parameter is a parameter p that is used only in guards and invariants of the form $p \prec x$ (resp. $x \prec p$), where x is a clock.*

2.4 Decision Problems

Let $\mathcal{P}$ be a given a class of decision problems (reachability, unavoidability, etc.).

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

$\mathcal{P}$-universality problem:
INPUT: A PTA $\mathcal{A}$ and an instance ϕ of $\mathcal{P}$
PROBLEM: Are all parameter valuations v such that $v(\mathcal{A})$ satisfies ϕ?

Emptiness is the most basic parametric question: is there at least one parameter valuation such that the property holds? Universality gives a robustness quality to the property and permits to effectively abstract an infinite number of verifications with concrete values.

In this paper, we mainly focus on reachability and unavoidability properties, and call them EF and AF respectively. For example, given a PTA $\mathcal{A}$ and a subset G of its locations, EF-universality asks: "are all parameter valuations v such that G is reachable in $v(\mathcal{A})$ from the initial state?" And AF-emptiness asks: "is the set of valuations v of the parameters such that G is unavoidable in $v(\mathcal{A})$ empty?"

3 Undecidability of EF-Emptiness

Let us first recall the following classical result for PTAs.

Theorem 1 ([Mil00]). *The EF-emptiness problem is undecidable for bounded PTAs.*

We provide an alternative and original proof of this result. This new construction is similar to that of Miller [Mil00], but it might be seen as a bit simpler and we will provide a complete proof. And above all, it allows us to extend it to obtain several of the main results of this paper.

Proof. We build a PTA that encodes a 2-counter machine (2CM) [Min67], such that the machine halts iff there exists some valuation of the parameters of the PTA such that it reaches a specific location.

Recall that such a machine has two non-negative counters C_1 and C_2, a finite number of states and a finite number of transitions, which can be of the form:

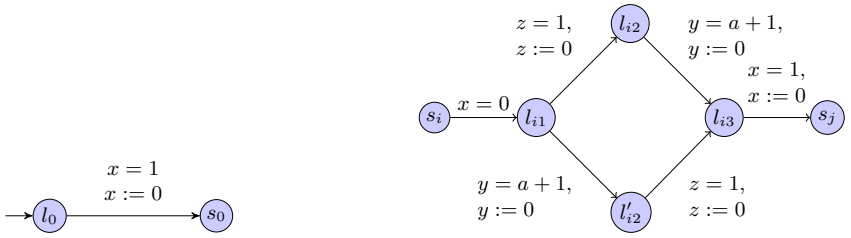
– when in state s_i, increment C_k and go to s_j;
– when in state s_i, decrement C_k and go to s_j;
– when in state s_i, if $C_k = 0$ then go to s_j, otherwise block.

The machine starts in state s_0 and halts when it reaches a particular state $l_{\texttt{halt}}$. The halting problem for 2-counter machines is undecidable [Min67].

Given such a machine $\mathcal{M}$, we now provide an encoding as a PTA $\mathcal{A}(\mathcal{M})$: each state s_i of the machine is encoded as a location of the automaton, which we also call s_i.

The counters are encoded using clocks x, y and z and one parameter a, with the following relations with the values c_1 and c_2 of counters C_1 and C_2: in any location s_i, when $x = 0$, we have $y = 1 - ac_1$ and $z = 1 - ac_2$. Note that all three clocks are parametric, *i. e.,* are compared with a in some guard or invariant. We will see that a is a rational-valued bounded parameter, typically in $[0, 1]$. The main idea of our encoding is that, to correctly simulate the machine, the parameter must be sufficiently small to encode the maximum value of the counters, *i. e.,* for $1-ac_1$ and $1-ac_2$ to stay non-negative all along the execution of the machine.

We initialize the clocks with the gadget in Figure 1a. Clearly, when in s_0 with $x = 0$, we have $y = z = 1$, which indeed corresponds to counter values 0.



(a) EF-emptiness: initial gadget

(b) EF-emptiness: increment gadget

Fig. 1: EF-emptiness: gadgets

We now present the gadget encoding the increment instruction of C_1 in Figure 1b.

The transition from s_i to l_{i1} only serves to clearly indicate the entry in the increment gadget and is done in 0 time unit.

Since we use only equalities, there are really only two paths that go through the gadget: one going through l_{i2} and one through l'_{i2}. Let us begin with the former.

We start from some encoding configuration: $x = 0$, $y = 1-ac_1$ and $z = 1-ac_2$ in s_i (and therefore the same in l_{i1}). We can enter l_{i2} (after elapsing enough time) if $1 - ac_2 \leq 1$, *i. e.,* $ac_2 \geq 0$, which implies that $a \geq 0$, and when entering l_{i2} we have $x = ac_2$, $y = 1 - ac_1 + ac_2$ and $z = 0$. Then we can enter l_{i3} if $1 - ac_1 + ac_2 \leq 1 + a$, *i. e.,* $a(c_1 + 1) \geq ac_2$. When entering l_{i3}, we then have

$x = a(c_1 + 1)$, $y = 0$ and $z = a(c_1 + 1) - ac_2$. Finally, we can go to s_j if $a(c_1 + 1) \leq 1$ and when entering s_j we have $x = 0$, $y = 1 - a(c_1 + 1)$ and $z = 1 - ac_2$, as expected.

We now examine the second path. We can enter l'_{i2} if $1 - ac_1 \leq a + 1$, *i. e.*,$a(c_1 + 1) \geq 0$, and when entering l'_{i2} we have $x = a(c_1 + 1)$, $y = 0$ and $z = 1 - ac_2 + a(c_1 + 1)$. Then we can go to l_{i3} if $1 - ac_2 + a(c_1 + 1) \leq 1 + a$, *i. e.*, $a(c_1 + 1) \leq ac_2$. When entering l_{i3}, we then have $x = ac_2$, $y = ac_2 - a(c_1 + 1)$ and $z = 0$. Finally, we can go to s_j if $ac_2 \leq 1$ and when entering s_j we have $x = 0$, $y = 1 - a(c_1 + 1)$ and $z = 1 - ac_2$, as expected.

Remark that exactly one path can be taken depending on the respective order of $c_1 + 1$ and c_2, except when both are equal or $a = 0$, in which cases both paths lead to the same configuration anyway.

Decrement is done similarly by replacing guards $y = a + 1$ with $y = 1$, and guards $x = 1$ and $z = 1$ with $x = a + 1$ and $z = a + 1$, respectively.

From s_i, to encode zero-testing C_1 and going to s_j, we only need to add a transition from s_i to s_j with guard $y = 1 \wedge x = 0$.

All those gadgets also work for C_2 by swapping y and z.

Finally, we add another location $l'_{\texttt{halt}}$ and a transition from $l_{\texttt{halt}}$ to $l'_{\texttt{halt}}$ with guard $0 < x < 1$ and $x = a$. This implies the constraint $0 < a < 1$ when reaching $l'_{\texttt{halt}}$. This is important, in order to remove the $a = 0$ value, which does not encode the counters properly. (Note that we could also do this as early as the initialization gadget; however, it is convenient to leave it here for the subsequent proofs reusing this proof.) Removing the value $a = 1$, which would be possible if both counters are always 0, is not necessary but it will be useful in subsequent proofs.

Let us now prove that the machine halts iff there exists a parameter valuation p such that $p(\mathcal{A})$ can reach $l'_{\texttt{halt}}$. Consider two cases:

1. Either the machine halts, then the automaton can go into the $l'_{\texttt{halt}}$ location, with constraints $0 < a < 1$ and, if c is the maximum value of both C_1 and C_2 over the (necessarily finite) halting run of the machine, and if $c > 0$, then $a \leq \frac{1}{c}$. The set of such valuations for a is certainly non-empty: $a = \frac{1}{2}$ belongs to it if $c = 0$ and $a = \frac{1}{c}$ does otherwise;
2. Or the machine does not halt. There are two subcases:
   (a) either the counters stay bounded. Let c be their maximal value. As before, if $c = 0$ and $0 < a \leq 1$ or $c > 0$ and $ca < 1$, then the machine is correctly encoded and the PTA cannot reach $l'_{\texttt{halt}}$. Otherwise, at some point during an incrementation of, say, C_1 we will have $a(c_1 + 1) > 1$ when taking the transition from l_{i2} to l_{i3} and the PTA will be blocked;
   (b) or one of the counters is not bounded, say C_1. Then whatever the value of $a > 0$, we have the same situation as in the previous item: the automaton blocks during some incrementation.

   In both subcases, the automaton cannot reach the $l'_{\texttt{halt}}$ location and the set of parameters such that it does is obviously empty.

□

Remark 1. We use guards with constraints $y = a + 1$ while our definition of PTAs, following [AHV93], only allows comparisons of a clock with a single parameter. Note however, and that will be true for all subsequent constructions, that transitions with $y = a + 1$ guards and $y := 0$ reset can be equivalently replaced by one transition with an $y = 1$ guard and a reset of some additional clock w, followed by a transition with a $w = a$ guard and the $y := 0$ reset (and similarly for x and z is the decrement gadget). This allows the proof to work without complex parametric expressions in guards and uses only one parametric clock and three normal clocks, with one parameter, matching the best known results with that respect [Mil00].

Now, by reusing the previous proof, we can show that the EF-emptiness problem is undecidable for closed bounded PTAs. To the best of our knowledge, this is an original result, as all existing results with bounded PTAs (*e. g.,* [Mil00,Doy07]) require strict inequalities.

Theorem 2. *The EF-emptiness problem is undecidable for closed bounded PTAs.*

Proof. See Appendix A. □

4 Integer-Points Parametric Timed Automata

In this section, we introduce integer-points parametric timed automata (IP-PTAs for short), *i. e.,* a subclass of PTAs in which any symbolic state contains at least one integer point (Section 4.1). Our first result is to prove the decidability of the EF-emptiness problem for bounded IP-PTAs (Section 4.2). Then, we compare IP-PTAs with L/U-PTAs and show that the class of bounded IP-PTAs is strictly larger than bounded L/U-PTAs with non-strict inequalities (Section 4.3). We then show that synthesis is intractable in practice, and that the same holds for bounded L/U-PTAs (Section 4.4). Finally, although we prove that the membership problem is undecidable for IP-PTAs, we exhibit a syntactic sufficient condition, that provides a new subclass of PTAs for which the EF-emptiness problem is decidable (Section 4.5).

4.1 The Class of IP-PTAs

Definition 4. *A PTA $\mathcal{A}$ is said to be an* integer points PTA *(in short* IP-PTA*) if, in any reachable symbolic state (l, C) of $\mathcal{A}$, C contains at least one integer point.*

4.2 A Decidability Result for Bounded IP-PTAs

Our main positive result is that the EF-emptiness problem is decidable for bounded IP-PTAs.

Theorem 3. *The EF-emptiness problem is decidable (and PSPACE-complete) for bounded IP-PTAs.*

Proof. We first need to recall two lemmas relating symbolic and concrete runs (proved in [HRSV02,ACEF09]).

Given a concrete (respectively symbolic) run $(l_0, \mathbf{0}) \overset{e_0}{\mapsto} (l_1, w_1) \overset{e_1}{\mapsto} \cdots \overset{e_{m-1}}{\mapsto} (l_m, w_m)$ (respectively $(l_0, C_0) \overset{e_0}{\Rightarrow} (l_1, C_1) \overset{e_1}{\Rightarrow} \cdots \overset{e_{m-1}}{\Rightarrow} (l_m, C_m)$), we define the corresponding discrete sequence as $l_0 \overset{e_0}{\Rightarrow} l_1 \overset{e_1}{\Rightarrow} \cdots \overset{e_{m-1}}{\Rightarrow} l_m$. Two runs (concrete or symbolic) are said to be equivalent if their associated discrete sequences are equal.

Lemma 1. *Let $\mathcal{A}$ be a PTA, and v be a parameter valuation. Let ρ be a run of $\mathcal{A}$ reaching a symbolic state (l, C). Then, there exists an equivalent run in the TA $v(\mathcal{A})$ reaching a concrete state (l, w) (for some w) iff $v \models C{\downarrow_P}$.*

Lemma 2. *Let $\mathcal{A}$ be a PTA, and v be a parameter valuation. Let ρ be a run of the TA $v(\mathcal{A})$ reaching a concrete state (l, w). Then there exists an equivalent run in $\mathcal{A}$ reaching a symbolic state (l, C), for some C such that $v \models C{\downarrow_P}$.*

Let $\mathcal{A}$ be a bounded IP-PTA. EF-emptiness is false for $\mathcal{A}$ iff there exists a valuation v such that a run of $v(\mathcal{A})$ reaches a location in some predefined set G. Assume there exists a valuation v such that a run of $v(\mathcal{A})$ reaches l, with $l \in G$. From Lemma 2, there exists a symbolic run of $\mathcal{A}$ reaching a symbolic state (l, C), for some C. Since $\mathcal{A}$ is an IP-PTA, C contains at least one integer point. Hence there exists an integer parameter valuation $v' \models C{\downarrow_P}$; hence from Lemma 1, there exists a concrete run of $v'(\mathcal{A})$ reaching l. This gives that EF-emptiness is false for $\mathcal{A}$ iff there exists an integer valuation v' such that a run of $v'(\mathcal{A})$ reaches a location in G.

Hence, deciding whether some valuation permits to reach l reduces to deciding whether some *integer* valuation permits to do so, which, for bounded PTAs, is PSPACE-complete [JLR15]. □

In practice, [JLR15] proposes efficient symbolic algorithms to synthesize all the integer parameter valuations that permit to reach some given location, and thus to solve EF-emptiness for IP-PTAs.

4.3 Comparison with L/U-PTAs

Let us now compare IP-PTAs and L/U-PTAs. We first need the following lemma, stating that any reachable symbolic state of an L/U-PTA contains an integer parameter valuation.

Lemma 3. *Let (l, C) be a reachable symbolic state of an L/U-PTA. Then $C{\downarrow_P}$ contains at least one integer point.*

Proof. Consider a (non-empty) reachable symbolic state (l, C) of an L/U-PTA. Let $v \models C{\downarrow_P}$. From the well-known monotonicity property of L/U-PTAs (exhibited in [HRSV02]), any parameter valuation such that the lower-bound parameters p_i^- are lower or equal to $v(p_i^-)$ and upper-bound parameters p_j^+ are greater than or equal to $v(p_j^+)$ also belong to $C{\downarrow_P}$. In particular, this is the case

of the integer parameter valuation assigning 0 to all lower-bound parameters, and assigning to upper-bound parameters p_j^+ the smallest integer greater than or equal to $v(p_j^+)$. □

The previous lemma that ensures the presence of an integer parameter valuation in any symbolic state does not guarantee that an L/U-PTA is an IP-PTA, because clocks may have non-integer values.

Proposition 1. *The class of IP-PTAs is incomparable with the class of L/U-PTAs.*

Proof.
- Consider an L/U-PTA with a transition guarded by $x > 0$ and resetting no clock, followed by a second location with invariant $x < 1$; then, necessarily, the symbolic state associated with this second location contains no integer point (as $x \in (0, 1)$ in that symbolic state).
- It is easy to exhibit an IP-PTA that is not an L/U-PTA. This is for example the case of a simple PTA with only one location, one clock x and one parameter p with a self-loop with guard $x = p$ and resetting x.

□

However, we can prove that any *closed* L/U-PTA, *i. e.,* with only non-strict inequalities, is an IP-PTA. In order to show that the class of closed L/U-PTAs is included in IP-PTAs, we need the following lemma.

Lemma 4. *Let $\mathcal{A}$ be a PTA with only non-strict inequalities. Let $\mathbf{s} = (l, C)$ be a symbolic state of $\mathcal{A}$. Then if $C{\downarrow_P}$ contains at least one integer parameter valuation, then C contains an integer point.*

Proof. Since there is at least one integer parameter valuation v in $C{\downarrow_P}$, then $v(C)$ is not empty. Since v is an integer valuation, $v(C)$ is a zone of a timed automaton with integer constants, so the vertices of $v(C)$ are integer points. Finally, there is at least one vertex in $v(C)$ because all clocks are nonnegative (and hence are bounded from below by 0), and this vertex does belong to $v(C)$ because it is topologically closed due to the non-strict constraints. So C contains at least one integer point. □

Proposition 2. *The class of IP-PTAs is strictly larger than the class of closed L/U-PTAs.*

Proof. From Lemmas 3 and 4, and Proposition 1 ($\Leftarrow$). □

The previous result also holds for bounded PTAs:

Proposition 3. *The class of bounded IP-PTAs is strictly larger than the class of closed bounded L/U-PTAs.*

Proof. Lemma 3 extends to bounded L/U-PTAs, since the bounds are integers (this would not hold otherwise). Then, the proof of Proposition 1 ($\Leftarrow$) holds with bounded IP-PTAs and closed bounded L/U-PTAs. Applying Lemma 4 concludes the proof. □

Proposition 4. *The class of bounded IP-PTAs is incomparable with the class of bounded L/U-PTAs. The class of bounded IP-PTAs is incomparable with the class of L/U-PTAs.*

Proof. The proof of Proposition 1 can be applied with bounded PTAs on either side. □

Since bounded IP-PTAs are incomparable with L/U-PTAs (for which the EF-emptiness problem is known to be decidable), and since L/U-PTAs are the only non-trivial subclass of PTAs for which this problem is known to be decidable, then Theorem 3 strictly extends the subclass of PTAs for which this problem is decidable.

4.4 Intractability of the Synthesis

Although the EF-emptiness problem is decidable for L/U-PTAs [HRSV02], the synthesis seems to pose practical problems: it was shown in [JLR15] that the solution to the EF-synthesis problem for L/U-automata, if it can be computed, cannot be represented using any formalism for which emptiness of the intersection with equality constraints is decidable. In particular, this rules out the possibility of computing the solution set as a finite union of polyhedra.

We reuse the intuition of this result and extend it to closed bounded L/U-PTAs.

Theorem 4. *If it can be computed, the solution to the EF-synthesis problem for closed bounded L/U-automata cannot be represented using any formalism for which emptiness of the intersection with equality constraints is decidable.*

Proof. We reuse the idea of [BL09] used for proving that constrained emptiness for infinite runs acceptance properties is undecidable, and reused in [JLR15, Theorem 2]. Suppose that the solution to the EF-synthesis problem for closed bounded L/U-PTAs can be represented using a formalism for which emptiness of the intersection with equality constraints is decidable. Assume a closed bounded PTA $\mathcal{A}$; for each parameter p_i of $\mathcal{A}$ that is used both as an upper bound and a lower bound, replace its occurrences as upper bounds by a fresh parameter p_i^u and its occurrences as lower bounds by a fresh parameter p_i^l. We therefore obtain a closed bounded L/U-PTA. Assume we can derive a solution to the EF-synthesis problem for this closed bounded L/U-PTA, and let K be that solution. Then, by hypothesis, we can decide whether $K \wedge \bigwedge_i p_i^l = p_i^u$ is empty or not; hence, we can solve the EF-emptiness for $\mathcal{A}$, which contradicts the undecidability of EF-emptiness for closed bounded PTAs (from Theorem 2). □

Corollary 1. *If it can be computed, the solution to the EF-synthesis problem for IP-PTAs cannot be represented using any formalism for which emptiness of the intersection with equality constraints is decidable.*

Proof. From the fact that a closed bounded L/U-PTA is an IP-PTA. □

4.5 Membership

We first show that it cannot be decided in general whether a PTA is a (bounded) IP-PTA.

Theorem 5. *It is undecidable whether a PTA is an IP-PTA, even when bounded.*

Proof. Let us consider the PTA $\mathcal{A}(\mathcal{M})$ encoding the 2-counter machine $\mathcal{M}$ proposed in our proof of Theorem 1. The PTA $\mathcal{A}(\mathcal{M})$ has only one parameter a and all the symbolic states of $\mathcal{A}(\mathcal{M})$ contain the integer value $a = 0$ except the states corresponding to the location $l'_{\texttt{halt}}$. Since all constraints are non-strict, except that of the transition leading to $l'_{\texttt{halt}}$, all reachable symbolic states, except those associated with $l'_{\texttt{halt}}$, contain an integer point. Then the PTA $\mathcal{A}(\mathcal{M})$ reaches the location $l'_{\texttt{halt}}$ if and only if $\mathcal{A}(\mathcal{M})$ is not an IP-PTA. As a consequence, this PTA is an IP-PTA iff the 2-counter machine does not halt. Finally, note that this PTA can be bounded by $0 \leq a \leq 1$, without any change in the reasoning above. □

Nevertheless, Proposition 2 provides a sufficient syntactic membership condition, since any closed L/U-PTA is an IP-PTA. In addition, we now define another new non-trivial set of restrictions leading to IP-PTAs:

Definition 5 (Reset-PTA). *A* reset-PTA *is a PTA where:*

- *all guards and invariants are conjunctions of constraints of the form* $x \leq p + k$, $x \geq p + k$, $x \leq k$, *or* $x \geq k$, *with* x *a clock,* p *a parameter, and* k *an integer;*
- *and all clocks are reset to 0 on any transition with a guard or a source location invariant in which a parameter appears.*

This kind of restriction is somewhat reminiscent of those enforced by *initialized* hybrid automata [HKPV98] to obtain decidability. We now prove that reset-PTAs are IP-PTAs, which in turn means that the EF-emptiness problem is decidable for bounded reset-PTAs. It is worth noting that, to the best our knowledge, bounded reset-PTAs and L/U-PTAs are the only non-trivial sets of syntactic restrictions of PTAs making the reachability emptiness problem decidable.

Theorem 6. *Any reset-PTA is an IP-PTA.*

Proof. See Appendix B. □

Recall that the synthesis is intractable for bounded IP-PTAs (from Corollary 1) and for bounded L/U-PTAs. In contrast, and although studying reset-PTAs in detail goes beyond the scope of this work, we highly suspect that exact synthesis can be computed for reset-PTAs (see remarks in Section 6).

5 New (Un)decidability Results for PTAs

In this section, we take advantage of the newly introduced class of IP-PTAs to solve several open problems on the more general class of PTAs; these results allow us to draw a better cartography of these problems for several subclasses of PTAs.

5.1 Undecidability of EF-Universality

We show below that, unlike L/U-PTAs, the EF-universality problem is undecidable for IP-PTAs even bounded. This result differentiates the classes of (bounded) L/U-PTAs and bounded IP-PTAs, and helps to understand better the boundary between decidability and undecidability for subclasses of PTAs.

Theorem 7. *The EF-universality problem is undecidable for bounded IP-PTAs.*

Proof. See Appendix C. □

Corollary 2. *The EF-universality problem is undecidable for IP-PTAs, for bounded PTAs, and for PTAs.*

Proof. From Theorem 7 and from the fact that a bounded IP-PTA is an IP-PTA, is a bounded PTA, and is a PTA. □

5.2 Undecidability of AF-Emptiness

It is known that AF-emptiness is undecidable for L/U-PTAs [JLR15]; reusing the encoding of the 2-counter machine proposed in our proof of Theorem 1, we now show that this result holds even for bounded L/U-PTAs.

Theorem 8. *The AF-emptiness problem is undecidable for bounded L/U-PTAs.*

Proof. See Appendix D. □

Corollary 3. *The AF-emptiness problem is undecidable for bounded IP-PTAs, for IP-PTAs and for bounded PTAs.*

Proof. The AF-emptiness problem is undecidable for bounded L/U-PTAs (Theorem 8), which immediately gives the undecidability for bounded PTAs.

Furthermore, the PTA used in the proof of Theorem 8 only uses non-strict inequalities; furthermore, $a^- = 0$ and $a^+ = 1$ is a parameter valuation solution of any symbolic state. Hence, from Lemma 4, this PTA is a bounded IP-PTA, which gives the result for bounded IP-PTAs. As a consequence, the result also holds for general IP-PTAs. □

Class	bL/U-PTAs	bIP-PTAs	L/U-PTAs	IP-PTAs	bPTAs	PTAs
EF-empt.	**Th. 10**	**Th. 3**	**[HRSV02]**	Th. 9	[Mil00]	[AHV93]
EF-univ.	**Th. 10**	Th. 7	**[BL09]**	Cor. 2	Cor. 2	Cor. 2
AF-empt.	Th. 8	Cor. 3	[JLR15]	Cor. 3	Cor. 3	[JLR15]

Table 1: Decidability results for PTAs and some subclasses

5.3 Summary

Before being able to summarize our results in Table 1, we need to prove two further missing results.

Theorem 9. *The EF-emptiness problem is undecidable for IP-PTAs.*

Proof. The proof of the undecidability of the EF-emptiness problem for general PTAs in [AHV93] can be interpreted over integer parameter valuations. Any symbolic state contains at least one integer parameter valuation (the one that is large enough to correctly encode the value of the two counters), as well as all larger parameter valuations. Furthermore, since the proof only uses non-strict inequalities (in fact only equalities), from Lemma 4, all symbolic states contain at least one integer point. Hence the PTA used in [AHV93] to encode the 2-counter machine is an IP-PTA. □

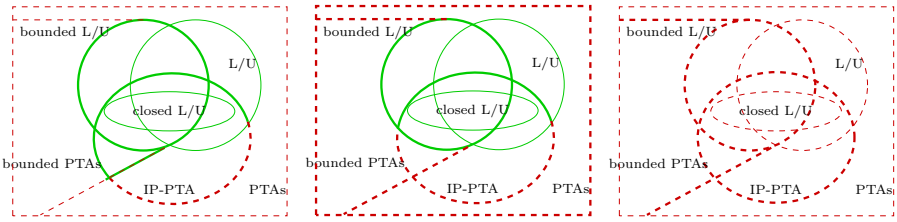
Finally, we show below (without surprise) that the EF-emptiness problem (shown to be decidable for L/U-PTAs [HRSV02]) and the EF-universality problem (shown to be decidable for integer-valued L/U-PTAs [BL09]) are also decidable for bounded L/U-PTAs.

Theorem 10. *The EF-emptiness and EF-universality problems are decidable for bounded L/U-PTAs.*

Proof. In [HRSV02,BL09], it is shown that decreasing a lower-bound parameter p_i^- or increasing an upper-bound parameter p_j^+ in an L/U-PTA $\mathcal{A}$ can only add behaviors. Hence, deciding EF-emptiness can be done by testing the reachability of the location in the TA obtained from $\mathcal{A}$ by instantiating all p_i^-s with 0 and all p_j^+s with ∞. (Recall that testing the reachability of a location in a TA is decidable [AD94].) For a bounded L/U-PTA, this can be done in a similar manner, by testing the reachability of the location in the TA obtained from $\mathcal{A}$ by instantiating all p_i^-s with their minimal value and all p_j^+s with their maximal value in the (closed) bounded parameter domain.

EF-universality can be solved similarly, except that p_i^-s are replaced with their upper bound and p_j^+s are replaced with their lower bound. □

We give a summary in Table 1. We give from left to right the (un)decidability for bounded L/U-PTAs, bounded IP-PTAs, L/U-PTAs, IP-PTAs, bounded PTAs, and PTAs. Decidability is given in bold green, whereas undecidability is given in thin red. Our contributions are depicted using a plain background, whereas existing results are depicted using a light background.



(a) EF-emptiness (b) EF-universality (c) AF-emptiness

Fig. 2: Decidability results for PTAs and subclasses

We give another summary in Figure 2. Note that bounded L/U-PTAs and L/U-PTAs are in fact incomparable of terms of expressiveness [ALR16]; they are therefore not included into each other in the figures. Decidability (resp. undecidability) is depicted in plain green (resp. dashed red); open problems are depicted in dotted black. Our contributions are depicted in thick.

6 Conclusion

In this paper, we exhibited a new subclass of PTAs (namely bounded IP-PTAs) for which the EF-emptiness problem is decidable. By showing that bounded IP-PTAs are incomparable with L/U-PTAs, we strictly extend the set of PTAs for which this problem is decidable. Although we showed that it cannot be decided whether a PTA is an IP-PTA, we introduced a new syntactic subclass of IP-PTAs, namely reset-PTAs, for which, when bounded, the EF-emptiness problem is decidable. It is worth noting that, to the best our knowledge, there is no other non-trivial set of syntactic restrictions making the reachability emptiness problem decidable for PTAs (aside from L/U-PTAs, of course).

In a second part, we considered three decision problems, and contributed in solving several open problems for PTAs and subclasses: this was achieved thanks to the results proved for IP-PTAs, and to (variations of) an original proof for the undecidability of the EF-emptiness problem for general PTAs with a single bounded rational-valued parameter and only non-strict constraints.

Future works Our new class of reset-PTAs seems promising in terms of synthesis, as the symbolic states have a very special form. Using a proper extrapolation, exact synthesis might be achievable. In addition, we are interested in extending this class to hybrid systems, and combining its restrictions with the condition of initialized hybrid automata [HKPV98]. The AF-universality problem is not treated in this paper, as it connects in an interesting manner with the problems of the existence of deadlocks or livelocks, which warrants a study on its own: in [AL16], we show in particular that the AF-universality problem is decidable for bounded L/U-PTAs with a closed parameter domain, and becomes undecidable if we lift either the assumption of boundedness or of closedness. Finally, all problems undecidable for L/U-PTAs remain open for L-PTAs and U-PTAs.

References

ACEF09. Étienne André, Thomas Chatain, Emmanuelle Encrenaz, and Laurent Fribourg. An inverse method for parametric timed automata. *International Journal of Foundations of Computer Science*, 20(5):819–836, 2009.

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

AHV93. Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. Parametric real-time reasoning. In *STOC*, pages 592–601. ACM, 1993.

AL16. Étienne André and Didier Lime. Liveness in L/U-parametric timed automata. Submitted. https://hal.archives-ouvertes.fr/hal-01304232, 2016.

ALR15. Étienne André, Didier Lime, and Olivier H. Roux. Integer-complete synthesis for bounded parametric timed automata. In *RP*, volume 9058 of *Lecture Notes in Computer Science*. Springer, 2015.

ALR16. Étienne André, Didier Lime, and Olivier H. Roux. On the expressiveness of parametric timed automata. In *FORMATS*, Lecture Notes in Computer Science. Springer, 2016. To appear.

And15. Étienne André. What's decidable about parametric timed automata? In *FTSCS*, volume 596 of *Communications in Computer and Information Science*, pages 1–17. Springer, 2015.

BBLS15. Nikola Beneš, Peter Bezděk, Kim G. Larsen, and Jiří Srba. Language emptiness of continuous-time parametric timed automata. In *ICALP, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 69–81. Springer, 2015.

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.

BY04. Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In *Lecture Notes on Concurrency and Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*. Springer–Verlag, 2004.

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

HKPV98. T.A. Henzinger, P.W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? *Journal of Computer and System Sciences*, 57:94–124, 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. *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 *HSCC*, volume 1790 of *Lecture Notes in Computer Science*, pages 296–309. Springer, 2000.

Min67. Marvin L. Minsky. *Computation: finite and infinite machines*. Prentice-Hall, Inc., NJ, USA, 1967.

Sch86. Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, Inc., New York, NY, USA, 1986.

A Proof of Theorem 2

Theorem 2 (recalled). *The EF-emptiness problem is undecidable for closed bounded PTAs.*

Proof. The entire encoding of the instructions of the 2-counter-machine used in the proof of Theorem 1 is a closed PTA, as only non-strict inequalities are used. In addition, the (unique) parameter a can be bounded by $[0, 1]$. However, the transition from $l_{\texttt{halt}}$ to $l'_{\texttt{halt}}$ uses strict inequalities in order to ensure that $0 < a < 1$. First, in contrast to Theorem 1, let us not remove $a = 1$, *i. e.*, the value that encodes the situation when both counters are always zero. (Recall that removing this value is not necessary, and that it was added to be used in the subsequent proofs working as variations of the proof of Theorem 1.) Now, removing $a = 0$ is necessary, as this valuation does not correctly encode the machine. Let us rewrite the transition from $l_{\texttt{halt}}$ to $l'_{\texttt{halt}}$ as in Figure 3.

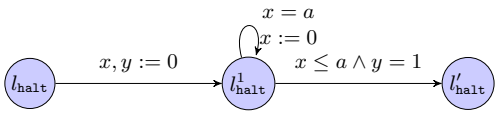


Fig. 3: Rewriting the last transition of the 2CM encoding with non-strict inequalities

Clearly, if $a = 0$, taking the self-loop on $l^1_{\texttt{halt}}$ will not allow time to elapse; and then there will be no way to leave $l^1_{\texttt{halt}}$ with $x \leq a$ and $y = 1$; hence $l'_{\texttt{halt}}$ is not reachable for $a = 0$. In contrast, if $0 < a \leq 1$, then by taking an appropriate number of times the self-loop on $l^1_{\texttt{halt}}$, we will eventually have $x \leq a$ and $y = 1$; hence $l'_{\texttt{halt}}$ will eventually be reached. To summarize:

- if $a = 0$, the machine is not correctly encoded, but there is no way to reach $l'_{\texttt{halt}}$;
- if $0 < a \leq 1$, the machine is correctly encoded, and from Theorem 1 we know that $l_{\texttt{halt}}$ is reachable iff the machine halts. Since $l'_{\texttt{halt}}$ is reachable from $l_{\texttt{halt}}$, then $l'_{\texttt{halt}}$ is reachable iff the machine halts.

Hence there exists a parameter valuation such that $l'_{\texttt{halt}}$ is reachable iff the machine halts.

B Proof of Theorem 6

Theorem 6 (recalled). *Any reset-PTA is an IP-PTA.*

Proof. We prove by induction that the symbolic states generated by reset-PTAs are zones with only non-strict constraints over the set of variables defined by the

union of clocks and parameters. To simplify the proof a bit we omit invariants but including them would raise no theoretical difficulty. We then additionally prove as part of the induction that there is no inequality involving two variables x and y in which x and y would not be of the same type (clock or parameter).

The property clearly holds for the initial symbolic state: parameters are unconstrained, all clocks are equal and their common value is greater than or equal to 0.

Now suppose this holds for some symbolic state and consider the successor of that symbolic state by some transition. Recall from the Succ definition that this successor is computed by the following operations: intersection with the guard of the transition, reset of the clocks designated in the reset set of the transition, and finally time elapsing.

Due to the restriction on constraints, all guards are themselves zones, and it is well-known that the intersection of two zones is again a zone. Similarly, the reset operation on some of the variables of a zone again leads to a zone. The time elapsing of a proper subset of the variables in a zone however is not a zone in general (the same situation arises, *e. g.,* in stopwatch automata). We therefore need to examine more closely the zone on which the time elapsing operates. Two cases arise:

1. the guard did not involve any parameter. Then, with the induction hypothesis, we still do not have any constraint between clocks and parameters after the intersection with the guard. A fortiori, we do not have any after the resets either. The time elapsing operation can be carried out by introducing a fresh non-negative variable t, performing variable substitutions $x \leftarrow x + t$ for each clock x, and finally eliminating t. This elimination can be done with the Fourier-Motzkin procedure (see *e. g.,* [Sch86]) which produces all the constraints not involving t plus those obtained by writing all the combinations of a minorant of t less or equal to a majorant of t. After the variable substitutions, t does not appear in constraints between parameters, nor in diagonal clock constraints ($x - y \leq k$ gives $(x + t) - (y + t) \leq k$, *i. e.,* again $x - y \leq k$). Since there is no constraint between clocks and parameters, t only appears in rectangular constraints that become of the form $y + t \leq k_1$ or $x + t \geq k_2$. Through the elimination procedure this gives constraints of the form $x - y \leq k_2 - k_1$, and the expected result holds.
2. the guard involves some parameters. Then before the reset we do have constraints between clocks and parameters. But then, from the definition of reset-PTAs, all clocks are reset along this transition, so these constraints are removed (as part of the elimination of clock variables) and replaced by constraints restricting the reintroduced clock variables to zero. Then, after the reset we do not have constraints between clocks and parameters anymore and the previous reasoning is again valid.

We conclude the proof by noting that in any non-empty zone with integer coefficients all vertices are integer (see the discussion in [JLR15]). And following the proof of Lemma 4, since all variables are non-negative, there is at least one

such vertex, which does belong to the zone because all constraints are non-strict. . This zone therefore contains at least an integer point. □

C Proof of Theorem 7

Theorem 7 (recalled). *The EF-universality problem is undecidable for bounded IP-PTAs.*

Proof. We start the encoding in our proof of Theorem 1. The main idea is, for all valuations of the parameter a that are not small enough to properly encode the counters (*i. e.,* for some value c of a counter, $1 - ac < 0$), to allow the PTA to directly go to an $l_{\texttt{error}}$ location. In order for our encoding to be an IP-PTA (in particular the $l_{\texttt{error}}$ symbolic states), we add a new parameter b, the value of which can be typically in $[0, 1]$.

We then reduce the problem of knowing whether the counters of the machine grow unbounded along its execution, which is undecidable [Min67], to the universality of the set of parameters that allow the encoding PTA to reach $l_{\texttt{error}}$.

First, we remove the $l'_{\texttt{halt}}$ location and the associated transition from $l_{\texttt{halt}}$, because it is no longer needed in this case and prevents the PTA to be an IP-PTA.

Instead, we create a fresh location $l_{\texttt{error}}$, and we add two transitions from l_0 (the initial location of the PTA) to the $l_{\texttt{error}}$ location:

- one with guard $x = 0 \wedge x = a$, that can only be taken when $a = 0$ and serves to "eliminate" this special case that does not correctly encode the counters.
- and one with guard $0 \leq x < 1 \wedge x = b$, that can only be taken when $b \in [0, 1)$.[4]

So, whenever $a = 0$ and $b \in [0, 1)$, the system can eventually reach $l_{\texttt{error}}$. Hence, in the following, we only need to focus on the case where $a \in (0, 1]$ and $b = 1$.

Let us now change the increment gadget (when decrementing or zero-testing, there is no upper bound constraint on a). More specifically, remark that, when incrementing C_1, the constraint that implies $a \leq \frac{1}{c_1+1}$ comes from the last transition in the path going through l_{i2}. In the other path, c_2 is already greater than or equal to c_1+1 and therefore a is already small enough to properly encode $c_1 + 1$ since it is small enough to encode c_2.

We modify the increment gadget as described in Figure 4.

In the transition from l_{i2} to l_{i3}, if a is not small enough then $x = a(c_1+1)$ will be greater than one and the final transition to s_j will not be fireable. We therefore add a direct transition from l_{i2} to $l_{\texttt{error}}$ when $x \geq b$. Recall that we only care about the case where $b = 1$, hence this transition can be understood as $x \geq 1$; now the case where $x = 1$ is problematic, as the value of a is just small enough to encode the counters, and we still can reach $l_{\texttt{error}}$, and the 2-counter machine is

[4] This case may not be necessary in the proof; however, it makes the explanation simpler, as we can now discard from our reasoning valuations such that $b \in [0, 1)$.

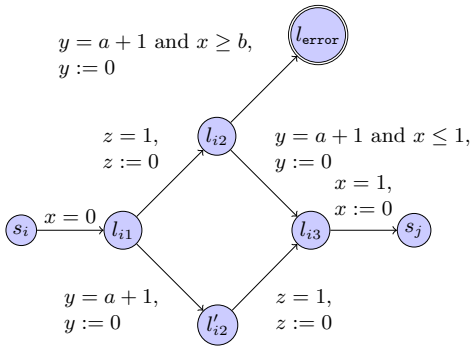


Fig. 4: EF-universality for bounded IP-PTAs: increment gadget

not properly encoded. However, since we are interested in universality, it suffices to take a valuation of a slightly smaller to properly encode the machine, as we explain more precisely below.

We now prove that the counters of the machine grow unbounded along its execution iff for all values of a and b, the encoding PTA can reach $l_{\texttt{error}}$. First recall that for $a = 0$ or $b \in [0, 1)$, it is always possible to reach $l_{\texttt{error}}$ (from the initial state). When $a > 0$ and $b = 1$, we have two cases:

- either the counters grow unbounded (say C_1 does), then whatever the value of $a > 0$, at some point we have $ac_1 > 1$. More specifically, there is an incrementation of C_1 such that $ac_1 \leq 1$ and $a(c_1 + 1) > 1$, which also implies $a(c_1 + 1) \geq b$ (since $b = 1$). Then, when executing the corresponding increment gadget, $l_{\texttt{error}}$ can be reached from l_{i2};
- or the counters stay bounded. Let c be the maximal value of the counters. Recall that when entering l_{i2} we have $x = ac_2$, $y = 1 - ac_1 + ac_2$ and $z = 0$. Then, when $y = a + 1$, we have $x = a(c_1 + 1)$. If $c_1 + 1 = c$, then $x = ca$ is the largest value that x can have in l_{i2} when $y = a+1$. Observe that, due to the non-strict inequality $x \geq b$ in the guard from l_{i2} to $l_{\texttt{error}}$, one might still reach $l_{\texttt{error}}$ for a valuation of a such that $ca = 1$. Consequently, consider the parameter valuation $a = \frac{1}{c+1}$ and $b = 1$. Then $ca < 1 = b$ and since x is in l_{i2} always at most equal to ca when $y = a + 1$, the guard to $l_{\texttt{error}}$ is never true and the of valuations for which the automaton can reach $l_{\texttt{error}}$ is not universal.

It remains to show that the constructed PTA is an IP-PTA. With the exception of $l_{\texttt{error}}$, the result is clear: $a = 0$ and $b = 0$ belongs to every reachable symbolic state, hence each symbolic state contains an integer parameter valuation, and hence from Lemma 4, all symbolic states (except $l_{\texttt{error}}$) contain at least one integer point. In addition, the two symbolic states taken by taking the two special transitions from the initial state to $l_{\texttt{error}}$ to handle $a = 0$ or $b \in [0, 1)$ also contain the integer point $x = y = a = b = 0$. Now, let us consider the other symbolic states with location $l_{\texttt{error}}$ (and reachable from some location l_{i2} due to

an increment). The projection onto the parameters of the associated constraint is $0 \leq b \leq 1 \wedge \frac{b}{i+1} \leq a \leq \frac{1}{i}$, with $i \in \mathbb{N}$ denotes the current maximum valuation of the counter. Clearly, $a = b = 0$ is a parametric integer point in this symbolic state; hence from Lemma 4 this symbolic state contains an integer point (in clocks and parameters dimensions). Hence this PTA is a (bounded) IP-PTA. □

D Proof of Theorem 8

Theorem 8 (recalled). *The AF-emptiness problem is undecidable for bounded L/U-PTAs.*

Proof. Let us consider the PTA $\mathcal{A}(\mathcal{M})$ encoding the 2-counter machine $\mathcal{M}$ proposed in our proof of Theorem 2. The PTA $\mathcal{A}(\mathcal{M})$ has only one parameter a which is used both as an upper bound and a lower bound. We add two fresh parameters a^- and a^+ and we replace the guard $y = 1 + a$ by a guard $1 + a^- \leq y$ and an invariant $y \leq 1 + a^+$ as shown for the increment gadget in Figure 5.

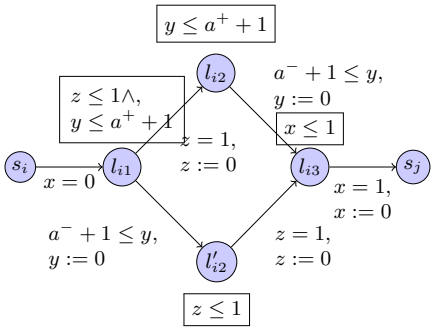


Fig. 5: AF-emptiness for bounded L/U-PTAs: increment gadget

We initialize the parameters a^- and a^+ with the gadget in Figure 6 (adapted from [JLR15]) leading to the location s_0. Clearly, starting from l_0, we have $\mathsf{AF}(s_0)$ if and only if $a^- = a^+ > 0$, because *1)* if $a^- = 0$ then it is possible to reach $l_{\texttt{sink}}$ and therefore we do not have $\mathsf{AF}(s_0)$, and *2)* any run that reaches l_1 before y is equal to a^+ can be extended by delaying a non-null amount of time into a run that will be blocked by the invariant of s_0. So all runs should enter l_1 with $y = a^+$, which is the case if and only if $a^- = a^+$. We therefore obtain an L/U-automaton with $a^- = a^+$ and $a^+ > 0$.

Let us now go back to the increment gadget of Figure 5. As in the proof of Theorem 2, when $a^- = a^+ > 0$, exactly one path can be taken depending on the respective order of $c_1 + 1$ and c_2.The invariants allow to avoid infinite delays in locations and since $a^- = a^+$, no run can be blocked inside the gadget. The same

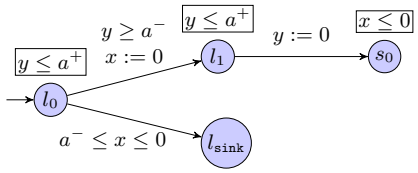


Fig. 6: AF-emptiness for bounded L/U-PTAs: initial gadget

reasoning can be made for the decrement and zero-testing gadgets. Moreover we can bound the PTA by $a^-, a^+ \in [0, 1]$ without loss of behavior.

Hence we reduce the halting problem of 2-counter machine to the AF-emptiness problem for bounded L/U-PTAs: the machine halts iff there exists a value of $a^- = a^+ > 0$, such that the location $l_{\texttt{halt}}$ is unavoidable in our bounded L/U-automaton. □