On the Expressiveness of Parametric Timed Automata

Étienne André, Didier Lime, Olivier Henri Roux

On the Expressiveness of Parametric Timed Automata*

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

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

Abstract. Parametric timed automata (PTAs) are a powerful formalism to reason about, model and verify real-time systems in which some constraints are unknown, or subject to uncertainty. In the literature, PTAs come in several variants: in particular the domain of parameters can be integers or rationals, and can be bounded or not. Also clocks can either be compared only to a single parameter, or to more complex linear expressions. Yet we do not know how these variants compare in terms of expressiveness, and even the notion of expressiveness for parametric timed models does not exist in the literature. Furthermore, since most interesting problems are undecidable for PTAs, subclasses, such as L/U-PTAs, have been proposed for which some of those problems are decidable. It is not clear however what can actually be modeled with those restricted formalisms and their expressiveness is thus a crucial issue. We therefore propose two definitions for the expressiveness of parametric timed models: the first in terms of all the untimed words that can be generated for all possible valuations of the parameters, the second with the additional information of which parameter valuations allow which word, thus more suitable for synthesis issues. We then use these two definitions to propose a first comparison of the aforementioned PTA variants.

Keywords: parametric timed automata, L/U-PTAs, hidden parameters

1 Introduction

Designing real-time systems is a challenging issue and formal models and reasoning are key elements in attaining this objective. In this context, timed automata (TAs) [AD94] are a powerful and popular modeling formalism. They extend finite automata with timing constraints, in which clocks are compared to integer constants that model timing features of the system. In the early design phases these features may not be known with precision and therefore parametric timed

* 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 FORMATS 2016. This author version also includes all proofs. The final publication is available at link.springer.com.

automata (PTAs) [AHV93] allow these constants to be replaced by unknown parameters, the correct values of which will be synthesized as part of the verification process. Unfortunately, most interesting problems are undecidable for PTAs, including the basic question of the existence of values for the parameters such that a given location is reachable [AHV93] (sometimes called EF-emptiness problem).

Since the seminal definition, many variants of PTAs have been defined in the literature, both as an effort to further increase the convenience of modeling by allowing complex linear expressions on parameters in the timing constraints (such as in [HRSV02,JLR15]), or in order to better assess the frontier of decidability for PTAs. In the latter objective, parameters have been considered to be integers [AHV93,Mil00,BL09,BO14,BBLS15,JLR15,AM15] or rationals [AHV93,Mil00,HRSV02,Doy07,JLR15,AM15], possibly bounded a priori [JLR15], or even restricted to be used as either always upper bounds or always lower bounds, giving so-called L/U-PTAs [HRSV02,BL09].

This difference in the class of constraints may have a direct impact on the decidability or complexity: for example, [BO14] recently improved the complexity of [AHV93] (NEXPTIME-complete instead of non-elementary) of the EF-emptiness problem over discrete time for one parametric clock and arbitrarily many non-parametric clocks and (integer-valued) parameters, but requires non-strict inequalities and uses invariants, features not used in the constructions of [AHV93]; it is hence unclear whether the result of [AHV93] is really subsumed by [BO14].

In order to be able to compare these definitions, one must first agree on a notion of expressiveness for timed parametric models, since none exists in the literature. This is the main objective of this work.

Contribution We propose the following two definitions of expressiveness: 1) as the union over all parameter valuations of the accepting untimed words ("untimed language"); 2) as the pairs of untimed words with the parameter valuations that allow them ("constrained untimed language").

We first prove that considering rational parameter valuations or unbounded integer parameter valuations in PTAs and L/U-PTAs is actually equivalent with respect to the untimed language.

We also prove that, whereas the untimed language recognized by a PTA with a single clock and arbitrarily many parameters is regular, adding a single non-parametric clock (i. e., a clock compared at least once to a parameter), even with a single parameter, gives a language that is at least context-sensitive, hence beyond the class of regular languages.

We then compare the expressiveness, w.r.t. untimed language and constrained untimed language, of several known subclasses of PTAs with integer parameters, in particular L/U-PTAs, and PTAs with bounded parameters. It turns out that, when considering the expressiveness as the untimed language, most subclasses of PTAs with integer parameters (including PTAs with bounded parameters, and L/U-PTAs) are in fact not more expressive than TAs. However, classical PTAs remain strictly more expressive than TAs. We also show that adding fully para-

metric constraints (i. e., comparison of parametric linear terms with 0, without any clock) does not increase the expressiveness of PTAs seen as the untimed language.

We also propose and focus on a new class of PTAs in which some parameters are hidden, i. e., do not occur in the constrained untimed language. While adding hidden parameters does not increase the expressiveness w.r.t. the untimed language (since in that case all parameters can be considered as hidden), when considering the expressiveness as the constrained untimed language, we show that hidden parameters strictly extend the expressiveness of PTAs. And interestingly, for this second definition of expressiveness, L/U-PTAs with bounded parameters turn out to be incomparable with classical L/U-PTAs.

Outline We introduce the basic notions in Section 2. We propose our two definitions of expressiveness in Section 3. We then show that rational-valued parameters are not more expressive than integer-valued parameters for the untimed language (Section 4). Focusing on integer-valued parameters, we then classify PTAs, their subclasses, and their extensions with hidden parameters w.r.t. the untimed language (Section 5) and the constrained untimed language (Section 6). We conclude and outline perspectives in Section 7.

2 Preliminaries

2.1 Clocks, Parameters and Constraints

Let $\mathbb{N}$, $\mathbb{Z}$, and $\mathbb{R}_+$ denote the sets of non-negative integers, integers, and non-negative real numbers respectively. Let $\mathcal{I}(\mathbb{N})$ denote the set of closed intervals on $\mathbb{N}$, i. e., the set of intervals $[a, b]$ where $a, b \in \mathbb{N}$ and $a \leq b$.

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 $\mu : X \to \mathbb{R}_+$. We write $\mathbf{0}$ for the clock valuation that assigns 0 to all clocks. Given $d \in \mathbb{R}_+$, $\mu + d$ denotes the valuation such that $(\mu + d)(x) = \mu(x) + d$, for all $x \in X$. Given $R \subseteq X$, we define the *reset* of a valuation μ, denoted by $[\mu]_R$, as follows: $[\mu]_R(x) = 0$ if $x \in R$, and $[\mu]_R(x) = \mu(x)$ otherwise.

We assume a set $P = \{p_1, \ldots, p_M\}$ of *parameters*, i. e., unknown integer-valued constants (except in Section 4 where parameters can also be rational-valued). A parameter *valuation* v is a function $v : P \to \mathbb{N}$.

In the following, we assume $\prec \in \{<, \leq\}$ and $\sim \in \{<, \leq, \geq, >\}$. Throughout this paper, 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 $\alpha_i, \beta_j, d \in \mathbb{Z}$. Similarly, plt denotes a parametric linear term over P, that is a linear term without clocks ($\alpha_i = 0$ for all i). A *constraint* C (i. e., a convex polyhedron) over $X \cup P$ is a conjunction of inequalities of the form $lt \sim 0$. 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 μ, $\mu(v(C))$ denotes the Boolean value obtained by replacing each clock x in $v(C)$ with $\mu(x)$.

A *guard* g is a constraint over $X \cup P$ defined by inequalities of the form $x \sim plt$.

2.2 Parametric Timed Automata with Hidden Parameters

Parametric timed automata (PTAs) extend timed automata with parameters within guards and invariants in place of integer constants [AHV93].

We actually first define an extension of PTAs (namely hPTAs) that will allow us to compare models with a different number of parameters, by considering that some of them are hidden. We will define PTAs as a restriction of hPTAs.

Definition 1 (PTA with hidden parameters). *A parametric timed automaton with hidden parameters (hereafter hPTA)* A *is a tuple* $(\Sigma, L, l_0, F, X, P, I, E)$*, 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)* $F \subseteq L$ *is a set of accepting locations, v)* X *is a finite set of clocks, vi)* $P = P_{\overline{v}} \uplus P_v$ *is a finite set of parameters partitioned into hidden parameters* $P_{\overline{v}}$ *and visible parameters* P_v*, vii)* I *is the invariant, assigning to every* $l \in L$ *a guard* $I(l)$*, viii)* E *is a finite set of edges* $e = (l, g, a, R, l')$ *where* $l, l' \in L$ *are the source and target locations,* $a \in \Sigma \cup \{\epsilon\}$ *(*ϵ *being the silent action),* $R \subseteq X$ *is a set of clocks to be reset, and* g *is a guard.*

We define a *PTA* as an hPTA in which $P = P_v$.

Observe that we allow ϵ-transitions (or silent transitions), i. e., transitions not labeled with any action.

Given an hPTA A and a parameter valuation v, we denote by $v(\mathsf{A})$ 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 an hPTA* $\mathsf{A} = (\Sigma, L, l_0, F, X, P, I, E)$*, and a parameter valuation* v*, the concrete semantics of* $v(\mathsf{A})$ *is given by the timed transition system* $(S, s_0, \rightarrow)$*, with* $S = \{(l, \mu) \in L \times \mathbb{R}_+^H \mid \mu(v(I(l)))$ *is true*$\}$*,* $s_0 = (l_0, \mathbf{0})$*, and* $\rightarrow$ *consists of the discrete and (continuous) delay transition relations:*

- *discrete transitions:* $(l, \mu) \xrightarrow{e} (l', \mu')$*, if* $(l, \mu), (l', \mu') \in S$*, there exists* $e = (l, g, a, R, l') \in E$*,* $\mu' = [\mu]_R$*, and* $\mu(v(g))$ *is true.*
- *delay transitions:* $(l, \mu) \xrightarrow{d} (l, \mu{+}d)$*, with* $d \in \mathbb{R}_+$*, if* $\forall d' \in [0, d], (l, \mu{+}d') \in S$*.*

A *(concrete) run* is a sequence $\rho = s_1 \alpha_1 s_2 \alpha_2 \cdots s_n \alpha_n \cdots$ such that $\forall i, (s_i, \alpha_i, s_{i+1}) \in \rightarrow$. We consider as usual that concrete runs strictly alternate delays d_i and discrete transitions e_i and we thus write concrete runs in the form $\rho = s_1 \xrightarrow{(d_1, e_1)} s_2 \xrightarrow{(d_2, e_2)} \cdots$. We refer to a state of a run starting from the initial state of a TA A as a *concrete state* (or just as a *state*) of A. Note that when a run is finite, it must end with a state. The *duration* of a concrete run is the sum of all the delays d_i appearing in this run.

An *untimed* run of $v(\mathsf{A})$ is a sequence $l_1 e_1 l_2 e_2 \cdots l_n \cdots$ such that for all i there exist a clock valuation μ_i and $d_i \geq 0$ such that $(l_1, \mu_1) \overset{(d_1,e_1)}{\rightarrow} (l_2, \mu_2) \overset{(d_2,e_2)}{\rightarrow} \cdots (l_n, \mu_n) \overset{(d_n,e_n)}{\rightarrow} \cdots$ is a run of $v(\mathsf{A})$. Given a run ρ, we denote by $\mathsf{Untime}(\rho)$ its corresponding untimed run.

The *trace* of an untimed run $l_1 e_1 l_2 e_2 \cdots l_n \cdots$ is the sequence $e_1 e_2 \cdots e_n \cdots$.

The *(untimed) trace* of a concrete run ρ is the trace of $\mathsf{Untime}(\rho)$.

A run ρ is *accepted* by $v(\mathsf{A})$ if it is finite and the location of its last state belongs to F. An untimed run is accepted by $v(\mathsf{A})$ if it is finite and its last location belongs to F.

The (untimed) language of $v(\mathsf{A})$ is the set of the traces of runs accepted by $v(\mathsf{A})$.

2.3 Subclasses of Parametric Timed Automata

L/U-PTAs have been introduced as a subclass of PTAs for which the EF-emptiness problem (i. e., the existence of values for the parameters such that a given location is reachable) is decidable [HRSV02]:

Definition 3 (hL/U-PTA). *An hL/U-PTA is an hPTA where the set of parameters is partitioned into a set of lower-bound parameters P^- and a set of upper-bound parameters P^+. A parameter p belongs to P^+ (resp. P^-), if it appears in constraints $x \leq plt$ or $x < plt$ always with a non-negative (resp. non-positive) coefficient, and in constraints $x \geq plt$ or $x > plt$ always with a non-positive (resp. non-negative) coefficient.*

Just as for PTAs, we define an *L/U-PTA* as an hL/U-PTA in which $P = P_v$.

Decidability comes from the fact that in L/U-PTAs increasing the value of an upper bound parameter or decreasing that of a lower bound parameter always only increase the possible behavior:

Lemma 1 (monotonicity of hL/U-PTAs [HRSV02]). *Let A be an hL/U-PTA and v be a parameter valuation. Let v' be a valuation such that for each upper-bound parameter p^+, $v'(p^+) \geq v(p^+)$ and for each lower-bound parameter p^-, $v'(p^-) \leq v(p^-)$. Then any run of $v(\mathsf{A})$ is a run of $v'(\mathsf{A})$.*

We will often use the notation p^+ (resp. p^-) for upper (resp. lower) bound parameters in hL/U-PTAs.

Given an hL/U-PTA, we denote by $v_{0/\infty}$ the special parameter valuation (mentioned in, e. g., [HRSV02]) assigning 0 to all lower-bound parameters and ∞ to all upper-bound parameters.[3]

Let us now define a bounded PTA as a PTA where the domain of each parameter is bounded, i. e., ranges between two integer-valued constants.

[3] Technically, $v_{0/\infty}$ is not a parameter valuation, as the definition of valuation does not allow ∞. However, we will use it only to valuate an L/U-PTA (or an hL/U-PTA) with it; observe that valuating an L/U-PTA with $v_{0/\infty}$ still gives a valid TA.

Definition 4 (bounded hPTA). *A* bounded hPTA *is* $\mathsf{A}_{|bounds}$*, where* A *is an hPTA, and bounds* $: P \to \mathcal{I}(\mathbb{N})$ *assigns to each parameter* p *an interval* $[\min, \max]$*, with* $\min, \max \in \mathbb{N}$*.*

We define similarly bounded hL/U-PTAs.

3 Defining the Expressiveness of PTAs

In the following, we denote by $\mathcal{V}(P)$, $\mathcal{V}(P_v)$, and $\mathcal{V}(P_{\overline{v}})$ the sets of valuations of respectively all the parameters, the visible parameters, and the hidden parameters of an hPTA.

Definition 5 (untimed language of an hPTA). *Given an hPTA* A*, the* untimed language *of* A*, denoted by* $\mathsf{UL}(\mathsf{A})$ *is the union over all parameter valuations* v *of the sets of untimed words accepted by* $v(\mathsf{A})$*, i. e.,*

$$\bigcup_{v \in \mathcal{V}(P)} \Big\{ w \mid w \textit{ is an untimed word accepted by } v(\mathsf{A}) \Big\}$$

TA is a subclass of PTA, hence, given a TA A, we also denote $\mathsf{UL}(\mathsf{A})$ its untimed language.

We propose below another definition of language for hPTAs, in which we consider not only the accepting untimed words, but also the parameter valuations associated with these words; this definition is more suited to compare the possibilities offered by parameter synthesis. Note that we only expose the *visible* parameter valuations.

Definition 6 (constrained untimed language of an hPTA). *Given an hPTA* A*, the* constrained untimed language *of* A*, denoted by* $\mathsf{CUL}(\mathsf{A})$ *is*

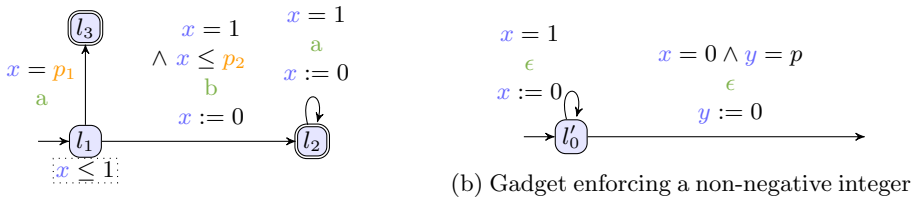
$$\bigcup_{v \in \mathcal{V}(P_v)} \Big\{ (w, v) \mid \exists v' \in \mathcal{V}(P_{\overline{v}}) \textit{ s.t. } w \textit{ is an untimed word accepted by } v(v'(\mathsf{A})) \Big\}$$

Note that since P_v and $P_{\overline{v}}$ are disjoint, we can write indifferently $v(v'(\mathsf{A}))$ and $v'(v(\mathsf{A}))$.

We use the word "constrained" because another way to represent the constrained language of an hPTA is in the form of a set of elements (w, K), where w is an untimed word, and K is a parametric constraint such that for all v in K, then w is an untimed word accepted by $v(v'(\mathsf{A}))$ for some $v' \in \mathcal{V}(P_{\overline{v}})$.

Example 1. Let us consider the hPTA A of Figure 1a, where $P_v = \{p_1\}$ and $P_{\overline{v}} = \{p_2\}$.

– Its untimed language is $\mathsf{UL}(\mathsf{A}) = \{a\} \cup \{ba^n \mid n \in \mathbb{N}\}$ that we note with the rational expression $\mathsf{UL}(\mathsf{A}) = a + ba^*$.



(a) A PTA

(b) Gadget enforcing a non-negative integer value for p

Fig. 1: An example of PTA, and a PTA gadget

– Its constrained untimed language is $\mathsf{CUL}(\mathsf{A}) = \left\{(a, p_1 = i) \mid 0 \leq i \leq 1\right\} \bigcup \left\{(ba^n, p_1 = i) \mid i \in \mathbb{N}, n \in \mathbb{N}\right\}$ that we can also note $\mathsf{CUL}(\mathsf{A}) = \left\{(a, p_1 \leq 1), (ba^*, p_1 \geq 0)\right\}$, with $p_1 \in \mathbb{N}$. Note that both the parameter p_2 and the fact that p_2 must be at least 1 to go to l_2 are hidden.

Definition 7 (regular constrained language). *The constrained untimed language of an hPTA* A *is* regular *if for all visible parameter valuations* $v \in \mathcal{V}(P_v)$*, the language* $\{w \mid (w, v) \in \mathsf{CUL}(\mathsf{A})\}$ *is regular.*

Remark 1. Since valuating a PTA with any rational parameter valuation gives a TA, the constrained untimed language of any PTA is regular in the sense of Definition 7.

Note that the idea of combining the untimed language with the parameter valuations leading to it is close to the idea of the behavioral cartography of parametric timed automata [AF10], that consists in computing parameter constraints together with a "trace set", i. e., the untimed language (that also includes in [AF10] the locations).

In the following, a *class* refers to an element in the set of TAs, bounded L/U-PTAs, L/U-PTAs, bounded PTAs and PTAs, and their counterparts with hidden parameters. An *instance* of a class is a model of that class.

A first class is strictly more expressive than a second one w.r.t. the untimed language if *i*) for any instance of the second one, their exists an instance of the first one that has the same untimed language, and *ii*) there exists an instance of the first one for which no instance of the second one has the same untimed language. Two classes are equally expressive w.r.t. the untimed language if for any instance of either class, their exists an instance of the other class that has the same untimed language. The comparison of the expressiveness w.r.t. the constrained untimed language can be defined similarly, with the additional requirement that the two instances must contain the same visible parameters (possibly after some renaming).

4 An Equivalence Between Integer and Rational Parameters

In the literature, some works focus on integer parameters [BO14,BBLS15,BL09], some others on rational parameters [HRSV02,Doy07], and also some propose constructions working in both settings [AHV93,Mil00,JLR15,AM15].

In this section, we prove that considering rational parameter valuations or unbounded integer parameter valuations in PTAs and L/U-PTAs is actually equivalent with respect to untimed languages.[4]

First, remark that any PTA with rational parameter valuations can be constrained to accept only non-negative integer parameter valuations. We just need to insert a copy of the gadget in Figure 1b for each parameter p before the initial location. We connect them to each other in sequence, in any order, and x and y can be clocks from the original PTA. In that gadget x is zero only when y is a non-negative integer and therefore p must be a non-negative integer to permit the exit from l'_0. Clearly, when considering only non-negative integer parameter valuations, both PTAs have the same untimed language.

With the above construction, we can filter out non-integer valuations. We can actually go a bit further and establish the following result:

Lemma 2. *For each PTA* A*, there exists a PTA* A' *such that:*

1. *for all rational parameter valuations* v *of* A *there exists an integer parameter valuation* v' *of* A' *such that* $v(\mathsf{A})$ *and* $v'(\mathsf{A}')$ *have the same untimed language.*
2. *for all integer parameter valuations* v' *of* A' *there exists a rational parameter valuation* v *of* A *such that* $v(\mathsf{A})$ *and* $v'(\mathsf{A}')$ *have the same untimed language.*

Proof. The idea of the proof is to scale all the expressions to which clocks are constrained so that they are integers. However, since we do not know in advance by how much we have to scale, we use an additional parameter to account for this scaling factor.

Let A be a PTA. Let p be a fresh parameter and let A'' be the PTA obtained from A by replacing every inhomogeneous (i. e., constant) term c in the linear expressions of guards and invariants by $c * p$. For instance, the constraint $x \leq 3p_1 + 2p_2 + 7$ becomes $x \leq 3p_1 + 2p_2 + 7p$.

We now build A' as follows: we add a new location (which will be the initial location of A'), from which two transitions, labeled ϵ and resetting all clocks, exit. The first one has guard $x \neq 0 \wedge x = p$ and goes to the initial location of A''. The second has guard $x = 0 \wedge x = p$ and goes to the initial location of an exact copy of A. By construction the first one can be taken only if $p \neq 0$ and the second one only if $p = 0$.

1. Let v be a rational parameter valuation of A. Let m be the least common multiple (LCM) of the denominators of the values assigned to parameters

[4] Comparing constrained languages would make no sense since obviously the parameter valuations cannot match in general in the rational and integer settings.

by v. Let v' be defined as: $\forall p_i \neq p, v'(p_i) = m * v(p_i)$ and $v'(p) = m$. Then, by construction, v' is an integer valuation of A', $v'(p) \neq 0$ and $v'(\mathsf{A}'')$ is a TA that is scaled by m from the TA $v(\mathsf{A})$. Then by [AD94, Lemma 4.1], $v(\mathsf{A})$ and $v'(\mathsf{A}'')$ have the same untimed runs up to renaming. And finally, $v(\mathsf{A})$ and $v'(\mathsf{A}')$ have the same untimed language.

2. The opposite direction works similarly: let v' be an integer parameter valuation of A'. If $v'(p) = 0$, then in A'' we can only go to the copy of A. We can therefore choose $v(p_i) = v'(p_i)$ and obtain the same untimed language. If $v'(p) \neq 0$, we define v by $v(p_i) = \frac{v'(p_i)}{v'(p)}$. Then v is a rational parameter valuation of A and $v(\mathsf{A})$ is a scaled down version of $v'(\mathsf{A}'')$, which therefore has the same untimed runs. And again, $v(\mathsf{A})$ and $v'(\mathsf{A}')$ have the same untimed language.

□

First remark that, in order to show the equivalence between integer- and rational-valued parameters, we provided a construction that added one additional parameter, and possibly some parametric clocks. This is consistent with the fact that PTAs with integer parameters typically have decidability results for slightly more parametric clocks and parameters than with rational parameters. For instance, the existence of a rational parameter valuation such that a given location is reachable is undecidable for PTAs with 1 parametric clock (a clock compared to parameters) and 3 normal clocks [Mil00], while the existence of an *integer* parameter valuation is decidable in that setting [BBLS15].

Second, in the construction, we need the integer parameters to be unbounded because the LCM can be arbitrarily big.

Finally, this result is not directly applicable to L/U-PTAs as we cannot ensure that the parameterized scaling factor would be the same for upper bound inhomogeneous terms as for lower bound ones. However, for L/U-PTAs, we can derive the same result from the monotonicity property:

Lemma 3. *For an L/U-PTA A, the set of untimed runs produced with only integer parameter valuations or with all rational parameter valuations is the same.*

Proof. Clearly the set of untimed runs produced by considering only integer parameter valuations is included in the one obtained by considering all rational parameter valuations.

In the other direction: let v be a rational parameter valuation of A and let v' be the integer parameter valuation obtained from v by rounding up the values for upper bound parameters, and rounding down for lower bound parameters. Then, by Lemma 1, $v'(\mathsf{A})$ contains all the untimed runs of $v(\mathsf{A})$. □

Here also we need integer parameters to be unbounded because the rational parameter valuations can themselves be arbitrarily big and we get accordingly big integers when rounding up.

We can now conclude the following:

Proposition 1. *PTAs (resp. L/U-PTAs) with rational parameters and PTAs (resp. L/U-PTAs) with unbounded integer parameters are equivalent with respect to the untimed language.*

When the parameters are bounded, we will see in Proposition 2 that the integer setting leads to regular languages. So, when bounded, PTAs with rational parameters are obviously strictly more expressive than their integer parameter counterpart. For L/U-PTAs, using again the monotonicity property, we trivially see that the valuation setting all upper-bound parameters to the maximal value allowed by the bounded domain, and lower-bound parameters to the minimal value gives all the untimed runs that are possible with other valuations. That "extremal" valuation is an integer valuation by definition. So, even when bounded, L/U-PTAs are still equally expressive in the rational and integer settings.

5 Expressiveness as the Untimed Language

5.1 PTAs in the Hierarchy of Chomsky

Let us show that (without surprise) Turing-recognizable languages (type-0 in Chomsky's hierarchy) can be recognized by PTAs (with enough clocks and parameters).

Lemma 4. *Turing-recognizable languages are also recognizable by PTAs.*

Proof. Consider a Turing-machine: it can be simulated by a 2-counter machine (with labelled instructions), which can in turn be simulated by a PTA. The transitions of the encoding PTA can be easily labeled accordingly (using also ϵ transitions). Assume that a word is accepted by the machine when it halts (i. e., it reaches $l_{\texttt{halt}}$). If the machine does not halt, $l_{\texttt{halt}}$ is reachable for no parameter valuation, hence the language of the machine is empty and that of the encoding PTA also. If the machine halts, $l_{\texttt{halt}}$ is reachable for parameter valuations correctly encoding the machine (i. e., depending on the proof, large enough or small enough to correctly encode the maximum value of the two counters). Hence, by taking the union over all parameter valuations of all untimed words accepted by the encoding PTA, one obtains exactly the language recognized by the machine. □

Lemma 4 only holds with enough clocks and parameters, typically 3 parametric clocks and 1 integer-valued or rational-valued parameter [BBLS15], or 1 parametric clock, 3 non-parametric clocks and 1 rational-valued parameter [Mil00].

For lower numbers, either decidability of the EF-emptiness problem is ensured (in which case the language cannot be type-0), or this problem remains open.

Let us point out a direct consequence of a result of [AM15] on PTAs with a single (necessarily parametric) clock.

Lemma 5. *The untimed language recognized by a PTA with a single clock and arbitrarily many parameters is regular.*

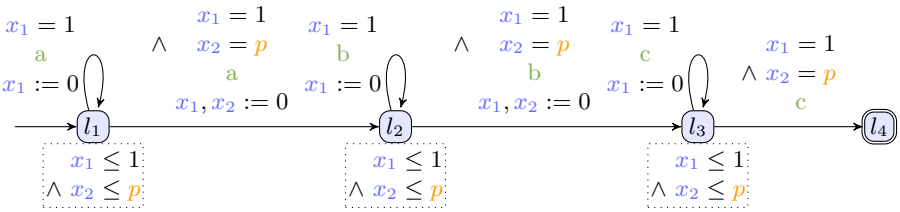


Fig. 2: A PTA with untimed language $a^n b^n c^n$

Proof. In [AM15, Theorem 20], we proved that the parametric zone graph (an extension of the zone graph for PTAs, following e. g., [JLR15]) of a PTA with a single (necessarily parametric) clock and arbitrarily many parameters is finite. This gives that the language recognized by a PTA with a single clock is regular. □

We now show that adding to the setting of Lemma 5 a single non-parametric clock, even with a single parameter, may give a language that is at least context-sensitive, hence beyond the class of regular languages.

Theorem 1. *PTAs with 1 parametric clock, 1 non-parametric clock and 1 parameter can recognize languages that are context-sensitive.*

Proof. Consider the PTA A in Figure 2. Consider an integer parameter valuation v such that $v(p) = i$, with $i \in \mathbb{N}$. The idea is that we use the parameter to first count the number of as, and then ensure that we perform an identical number of bs and cs; such counting feature is not possible in TAs (at least not for any value of i as is the case here). Clearly, due to the invariant $x_1 \leq 1$ in l_1, one must take the self-loop on l_1 every 1 time unit; then, one can take the transition to l_2 only after i such loops. The same reasoning applies to locations l_2 and l_3. Hence, the language accepted by the TA $v(\mathsf{A})$ is $a^{i+1}b^{i+1}c^{i+1}$.

Hence the union over all parameter valuations of the words accepted by A is $\{a^n b^n c^n \mid n \geq 1\}$. This language is known to be in the class of context-sensitive languages (type-1 in Chomsky's hierarchy), hence beyond the class of regular languages (type-3). □

This result is interesting for several reasons. First, it shows that adding a single clock, even non-parametric, to a PTA with a single clock immediately increases its expressiveness. Second, it falls into the interesting class of PTAs with 2 clocks, for which many problems remain open: the PTA exhibited in the proof of Theorem 1 (1 parametric clock and 1 non-parametric) falls into the class of 1 parametric clock, arbitrarily many non-parametric clocks and arbitrarily many integer-valued parameters, for which the EF-emptiness is known to be decidable [BBLS15]. When replacing the integer-valued with a rational-valued parameter (which does not fundamentally change our example), it also falls into the class of 1 parametric clock, 1 non-parametric clock and 1 rational-valued parameter, for which the EF-emptiness is known to be open [And15]. In both cases, it gives a lower bound on the class of languages recognized by such a PTA.

5.2 Comparison of Expressiveness

In this section, we compare the expressiveness of PTAs w.r.t. their untimed language UL.

First, we show in the following lemma that the untimed language of an L/U-PTA is equal to that of the same L/U-PTA valuated with $v_{0/\infty}$.

Lemma 6. *Let* A *be an L/U-PTA. Then:* $\mathsf{UL}(\mathsf{A}) = \mathsf{UL}(v_{0/\infty}(\mathsf{A}))$.

Proof. $\subseteq$ Let us first show that any accepting run of A for some parameter valuation is also an accepting run of $v_{0/\infty}(\mathsf{A})$, in the spirit of [HRSV02]. Let v be a parameter valuation. Let ρ be an accepting run of $v(\mathsf{A})$. Observe that, by definition, the guards and invariants of $v_{0/\infty}(\mathsf{A})$ are more relaxed than that of $v(\mathsf{A})$. Hence, any transition of ρ is also enabled in $v_{0/\infty}(\mathsf{A})$. Hence, ρ is also an accepting run of $v_{0/\infty}(\mathsf{A})$.

$\supseteq$ Conversely, let us show that, for any accepting run of $v_{0/\infty}(\mathsf{A})$, there exists a parameter valuation v such that this run is also an accepting run of $v(\mathsf{A})$. It suffices to show that, for a given run, there exists one parameter valuation accepting this run, as we define UL as the union over all parameter valuations. Let $\rho : s_0 \overset{(e_0,d_0)}{\to} s_1 \overset{(e_1,d_1)}{\to} \cdots \overset{(e_{m-1},d_{m-1})}{\to} s_m$ be an accepting run of $v_{0/\infty}(\mathsf{A})$. Let d be the duration of this run. Let $k = \lceil d \rceil + 1$. Let $v_{0/k}$ be the parameter valuation assigning 0 to all lower-bound parameters, and k to all upper-bound parameters. Now, observe that $v_{0/\infty}(\mathsf{A})$ and $v_{0/k}(\mathsf{A})$ are identical TAs, with the exception that some guards and invariants in $v_{0/k}(\mathsf{A})$ may include additional constraints of the form $x \leq i \times k$ or $x < i \times k$ (for some clock x and some $i > 0, i \in \mathbb{N}$). Since the duration of ρ is strictly less than k, then no clock will reach value k and therefore this run cannot be impacted by these additional constraints; hence, ρ is an accepting run of $v_{0/k}(\mathsf{A})$ too. □

Proposition 2. *TAs, L/U-PTAs and bounded PTAs are equally expressive w.r.t. the union of untimed languages.*

Proof. **L/U-PTAs = TAs** Direct from Lemma 6, and the fact that any TA is an L/U-PTA with no parameter.

bounded PTAs = TAs The untimed language of a PTA is the union of the untimed language of the TAs over all possible parameter valuations. As we consider integer-valued parameters, there is a finite number of valuations in a bounded PTA. Since the language recognized by a TA is a regular language, and the class of regular languages is closed under finite union, then bounded PTAs also recognize regular languages, and are therefore equally expressive with TAs.

□

Proposition 3. *L/U-PTAs and hL/U-PTAs are equally expressive w.r.t. the union of untimed languages.*

Proof. Consider an L/U-PTA A. Let A_h be the hL/U-PTA that is identical to A and contains no hidden parameters (i. e., $P_v = P$ and $P_{\overline{v}} = \emptyset$). Then $\mathsf{UL}(\mathsf{A}_h) = \mathsf{UL}(\mathsf{A})$.

Conversely, consider an hL/U-PTA A_h with visible parameters P_v and hidden parameters $P_{\overline{v}}$. Let A be the L/U-PTA such that $P = P_v \cup P_{\overline{v}}$. Then $\mathsf{UL}(\mathsf{A}) = \mathsf{UL}(\mathsf{A}_h)$. □

Proposition 4. *PTAs are strictly more expressive than TAs w.r.t. the union of untimed languages.*

Proof. Since the untimed words recognized by TA form a regular language [AD94], then the PTA exhibited in Theorem 1 recognizes a language not recognized by any TA. Conversely, any TA is a PTA (with no parameter) which gives that the expressiveness of PTAs is strictly larger than that of TAs.

□

In the following, we show that neither hidden parameters nor fully parametric linear constraints increase the expressive power of PTAs w.r.t. the union of untimed languages.

Proposition 5. *PTAs and hPTAs are equally expressive w.r.t. the union of untimed languages.*

Proof. Following the same reasoning as in Proposition 3. □

Impact of the syntax of the guards Recall that our guards and invariants are of the form $x \sim plt$, with plt a parametric linear term. Several alternative definitions exist in the literature. In addition to the PTAs defined in Definition 1, we consider here two other definitions, one that can be seen as the most restrictive (and used in e. g., [AHV93]), and one that is very permissive, with even constraints involving no clocks. We denote by a *simple guard* a constraint over $X \cup P$ defined by inequalities of the form $x \sim z$, where z is either a parameter or a constant in $\mathbb{Z}$. We define an *AHV93-PTA* as a PTA the guards and invariants of which are all conjunctions of simple guards. We define a PTA with fully parametric constraints (*fpc-PTA*) as a PTA the guards and invariants of which are conjunctions of inequalities either of the form $x \sim plt$ ("guards"), or $plt \sim 0$ ("fully parametric guards"). Let us show that all three definitions are equivalently expressive w.r.t. the untimed language.

Proposition 6. *PTAs and AHV93-PTAs are equally expressive w.r.t. the union of untimed languages.*

Proof. See Appendix A. □

This result extends in a straightforward manner to fpc-PTAs.

Proposition 7. *PTAs and fpc-PTAs are equally expressive w.r.t. the union of untimed languages.*

Proof. See Appendix B. □

(a) Bounding a PTA

(b) PTA accepting a for any valuation

Fig. 3: A PTA gadget and a PTA

6 Expressiveness as the Constrained Untimed Language

In this section, we compare the expressiveness of PTAs w.r.t. their visible constrained untimed language.

Proposition 8. *Bounded PTAs are strictly less expressive than PTAs w.r.t. the constrained untimed language.*

Proof. Bounded PTAs can easily be simulated using a non-bounded PTA, by bounding the parameters using one clock and appropriate extra locations and transitions prior to the original initial location of the PTA. For example, if x is reset when entering l'_1, the gadget in Figure 3a ensures that $p \in [\min, \max]$. All such gadgets (one per parameter) must be added in a sequential manner, resetting x prior to each gadget, and resetting all clocks when entering the original initial location after the last gadget.

Now, it is easy to find a PTA that has a larger constrained untimed language than any bounded PTA. This is the case of any PTA for which a word is accepting for parameter valuations arbitrarily large (e. g., Figure 3b). □

We now show that, interestingly, this result does not extend to L/U-PTAs, i. e., bounded L/U-PTAs are not strictly less expressive than but incomparable with L/U-PTAs.

Proposition 9. *Bounded L/U-PTAs are incomparable with L/U-PTAs w.r.t. the constrained untimed language.*

Proof.
- Let us show that the constrained untimed language of a given bounded L/U-PTA cannot be obtained for any L/U-PTA. Consider a bounded U-PTA with a single parameter p^+ with bounds such that $p^+ \in [0, 1]$, and accepting a for any valuation of $p^+ \in [0, 1]$. From Lemma 1, if this run is accepted in an L/U-PTA A', then this run is also accepted for any valuation v' such that $v'(p^+) \ge 0$, including for instance $v'(p^+) > 1$. Hence accepting a only for valuations of $p^+ \in [0, 1]$ cannot be obtained in an L/U-PTA, and therefore no L/U-PTA yields this constrained untimed language.
- This converse is immediate: assume an L/U-PTA with a single parameter p^+, accepting a for any valuation of $p^+ \in [0, \infty)$. From the definition of bounded (L/U-)PTAs, all parameters must be bounded, and therefore there exists no

bounded L/U-PTA that can accept a run for $p^+ \in [0, \infty)$. Hence no bounded L/U-PTA yields this constrained untimed language.

□

We now show that hidden parameters do not extend the expressiveness of L/U-PTAs (proof is in Appendix C).

Proposition 10. *hL/U-PTAs are equally expressive with L/U-PTAs w.r.t. the constrained untimed language.*

Hidden parameters however strictly extend the expressiveness of PTAs.

Lemma 7. *There exists an hPTA* A *such that* $\mathsf{CUL}(\mathsf{A})$ *is not regular.*

Proof. Assume a PTA with no parameter. Its constrained untimed language is a set of pairs (w, v), where v is a degenerate parameter valuation (i. e., a valuation $v : \emptyset \to \mathbb{N}$ as this PTA contains no parameter). The projection of this set of pairs onto the words (i. e., $\{w \mid (w, v) \in \mathsf{CUL}(\mathsf{A})\}$) yields a regular language, as a PTA without parameters is a TA, the class of language recognized by which is that of regular languages. Now consider an hPTA where all parameters are hidden. This time, from Theorem 1 the projection of its constrained untimed language onto the words yields a language that goes beyond the class of regular languages. Hence there exists an hPTA for which the constrained untimed language is not regular. □

Remark 2. The idea used in the proof of Lemma 7 uses a PTA with no (visible) parameter. But such a result can be generalized to a PTA with an arbitrary number of visible parameters: assume such a PTA, and assume one of its parameter valuations v. We can extend this PTA into a PTA A' with a single hidden parameter such that, for the valuation v (of the visible parameters), the PTA will produce $a^n b^n c^n$ using the construction in Theorem 1. Hence, the constrained untimed language of A' is not regular.

Proposition 11. *hPTAs are strictly more expressive than PTAs w.r.t. the constrained untimed language.*

Proof. From Remark 1 and Lemma 7. □

Let us finally show that PTAs and fpc-PTAs (involving additionally $plt \sim 0$) are not more expressive than AHV93-PTAs with hidden parameters.

Proposition 12. *PTAs and fpc-PTAs are not more expressive than AHV93-PTAs with hidden parameters w.r.t. the constrained untimed language.*

Proof. In Propositions 6 and 7, we used a construction to show the equivalent expressiveness of the untimed language of PTAs, fpc-PTAs and AHV93-PTAs. This construction transforms a PTA or an fpc-PTA into an AHV93-PTAs. Since we use extra parameters in this construction, it suffices to hide these extra parameters, and we therefore obtain an AHV93-PTA with the same CUL as the original (fpc-)PTA. □

Remark 3. In fact, we highly suspect that PTAs and fpc-PTAs are not more expressive than AHV93-PTAs (without hidden parameters). We cannot use our construction of Propositions 6 and 7 as it adds extra parameters, and therefore cannot ensure the equality of the CUL. However, we could propose an alternative construction using no extra parameter, at the cost of (many) extra parametric clocks (typically two per different guard in the PTA) and many extra locations. The idea would be to replace each guard with a gadget to be synchronized with the original PTA: this gadget will ensure that the sum of parameters is indeed achieved, using an extra clock counting each parameter. For negative parameter coefficients, this can be achieved thanks to another clock nondeterministically reset, and that must be equal to p_2 whenever the original clock x is reset. Typically, to ensure $x \sim p_1 - p_2$, we nondeterministically reset x_1, x_2; then whenever we reset x, then we must have $x_2 = p_2$. Finally, the guard $x \sim p_1 - p_2$ becomes $x_1 \sim p_1$. Proposing a formal construction is among our future works.

7 Conclusion and Perspectives

In this paper, we proposed a first attempt at defining the expressiveness of parametric timed automata, also introducing the notion of hidden parameters to compare models with different numbers of parameters. When considering the union over all parameter valuations of the untimed language, it turns out that all subclasses of PTAs with integer parameters are not more expressive than TAs. However, PTAs are strictly more expressive than TAs (from 1 parametric clock and 1 non-parametric clock); extending PTAs with hidden parameters or fully parametric constraints does not increase their expressiveness. In addition, integer-valued or rational-valued parameters turn out to be equivalent.

When considering the set of accepting untimed words together with their associated parameter valuations, then subclasses of PTAs with integer parameters have a varying expressiveness. An interesting result is that bounded L/U-PTAs turn out to be incomparable with L/U-PTAs. In addition, hidden parameters strictly extend the expressiveness of PTAs.

Future works We compared so far general formalisms; it now remains to be studied what consequences on decidability the forms of guards and invariants together with a fixed number of clocks and parameters may have: a ultimate goal would be to unify the wealth of (un)decidability results from the literature with all different syntactic contexts.

We showed that rational-valued parameters are not more expressive than integer-valued parameters; our construction makes use of an extra parameter. It remains to be shown whether this construction is optimal or not.

Finally, forbidding ϵ-transitions may also change our comparison of formalisms, as such silent transitions have an impact on the expressiveness of TAs (see, e. g., [BPDG98]).

References

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

AF10. Étienne André and Laurent Fribourg. Behavioral cartography of timed automata. In *RP*, volume 6227 of *Lecture Notes in Computer Science*, pages 76–90. Springer, 2010.

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

AM15. Étienne André and Nicolas Markey. Language preservation problems in parametric timed automata. In *FORMATS*, volume 9268 of *Lecture Notes in Computer Science*, pages 27–43. Springer, 2015.

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.

BO14. Daniel Bundala and Joël Ouaknine. Advances in parametric real-time reasoning. In *MFCS*, volume 8634 of *Lecture Notes in Computer Science*, pages 123–134. Springer, 2014.

BPDG98. Béatrice Bérard, Antoine Petit, Volker Diekert, and Paul Gastin. Characterization of the expressive power of silent transitions in timed automata. *Fundamenta Informaticae*, 36(2-3):145–182, 1998.

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

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.

Appendix

A Proof of Proposition 6

Proposition 6 (recalled). *PTAs and AHV93-PTAs are equally expressive w.r.t. the union of untimed languages.*

Proof. The first direction (AHV93-PTAs are not more expressive than PTAs) is trivial: any AHV93-PTA is a PTA.

Let us consider the other direction. We transform a PTA A into an equivalent AHV93-PTA, that will contain two additional parametric clocks (x_1, x_2, that can be existing clocks of the PTA) and several additional parameters (one per different parametric linear term).

Consider a constraint of the form $x \sim \sum_i \beta_i p_i$ appearing in a guard or an invariant. The idea is to create an additional parameter p, and ensure that $\sum_i \beta_i p_i = p$. We transform the equality $\sum_i \beta_i p_i = p$ into $\sum_j \beta_{i_j} p_{i_j} = \sum_k \beta_{i_k} p_{i_k} + p$, where $\beta_{i_j}, \beta_{i_k} \in \mathbb{N}$, that is we move to the right-hand side all negative coefficients so that they become positive. Then, we create the gadget as in Figure 4. Note that we make use of the shortcut $x_1 = \beta_i \times p_i$ for β_i consecutive transitions with guard $x_1 = p_i$ and resetting x_1 (recall that all our coefficients are now non-negative, since we moved the negative coefficient to the other side of the equality).

This gadget consists in the product of two PTA parts, that synchronize as follows: they start in their initial location (i. e., l_i^1 and l_i^2 respectively) with their respective clock (x_1 and x_2) equal to 0. Then, we synchronize the PTA parts through strong synchronization with renaming (via the a transitions), where the transitions to the respective final location (i. e., l_f^1 and l_f^2 respectively) can only be synchronized if both PTA parts are ready to take it together; note that they must reach the location preceding the a transition exactly at the same time due to the urgent invariants. We assume that a is a fresh label not used in the original PTA, and that the a label is renamed into ϵ when synchronized.

Through this synchronization, this ensures that the duration of the upper PTA part ($\sum_j \beta_{i_j} p_{i_j}$) is equal to that of the lower PTA part ($\sum_k \beta_{i_k} p_{i_k} + p$).

All these transitions are labeled with ϵ. At the end of this gadget, we necessarily have $p = \sum_i \beta_i p_i$.

A' is obtained from A as follows: first, we replace all occurrences in A of a constraint $x \sim plt$ with $x \sim p$ (where p is the additional parameter created when handling plt using the above construction). Second, we add before the initial location of A all necessary gadgets in a sequential manner, and we reset all clocks on the transition leading to the original initial location of A.

The initial location of A' is the initial location of the first gadget (i. e., one in the form of Figure 4). Since all transitions in the initial gadget are labeled with ϵ (recall that we use a synchronization with renaming so that the a transitions become ϵ when synchronized), the untimed language of A' is not impacted. □

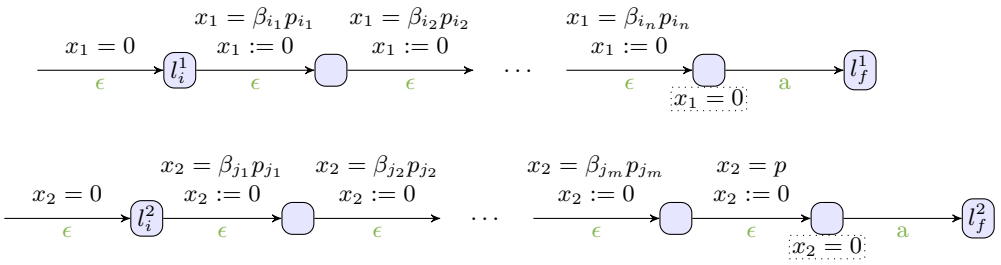


Fig. 4: Encoding a parametric constraint

B Proof of Proposition 7

Proposition 7 (recalled). *PTAs and fpc-PTAs are equally expressive w.r.t. the union of untimed languages.*

Proof. Fpc-PTAs may also contain constraints of the form $plt \sim 0$. The same construction as in Proposition 6 can be used, except that in the PTA the constraint $plt \sim 0$ is replaced with $p \sim 0$. Technically, the constraint $p \sim 0$ is not allowed in an AHV93-PTA; it can be simulated using an extra clock equal to 0, and to be compared with p. (Ensuring this clock is equal to 0 may require to duplicate the current location and to add an ϵ-transition.) □

Remark 4. The above construction requires two additional clocks and as many additional parameters as the number of (different) parametric constraints in the PTA. Clearly, our two additional clocks in the initial gadgets are not necessary, as the other clocks used in the PTA can be used instead. However, for fpc-PTAs, the clock used to simulate "$p \sim 0$" is necessary in our construction; and so seem to be the ϵ-transitions, the extra locations, and all extra parameters.

C Proof of Proposition 10

Proposition 10 (recalled). *hL/U-PTAs are equally expressive with L/U-PTAs w.r.t. the constrained untimed language.*

Proof. $\subseteq$ Let $\mathsf{A}_h = (\Sigma, L, l_0, F, X, P, I, E)$ be an hL/U-PTA, where $P = P_{\overline{v}} \uplus P_v$. Let $v^{\overline{p}}_{0/\infty}$ be the partial parameter valuation assigning 0 to any lower-bound parameter of $P_{\overline{v}}$ and ∞ to any upper-bound parameter of $P_{\overline{v}}$ (and not defined for parameters in P_v). As an abuse of notation, we denote by $v^{\overline{p}}_{0/\infty}(\mathsf{A}_h)$ the PTA where each occurrence of a parameter $p \in P_{\overline{v}}$ has been replaced with $v^{\overline{p}}_{0/\infty}(p)$. The obtained automaton is still parametric, as parameters in P_v are left untouched; also note that the obtained PTA contains no hidden parameter. We will show in the following that $\mathsf{CUL}(\mathsf{A}_h) = \mathsf{CUL}(v^{\overline{p}}_{0/\infty}(\mathsf{A}_h))$.

– Let $(w, v) \in \mathsf{CUL}(\mathsf{A}_h)$. Then there exists a valuation v' of the hidden parameters such that w is accepted by $v'(v(\mathsf{A}_h))$. By the monotonicity property ([HRSV02]) of the L/U-PTA $v(\mathsf{A}_h)$, replacing upper bounds in the parameters assigned by v' by $+\infty$ and lower bounds by 0 gives a TA in which w is still accepted: w is accepted by $v^{\overline{p}}_{0/\infty}(v(\mathsf{A}_h))$. Finally, the hidden and visible parameter sets being disjoint, this means w is accepted by $v(v^{\overline{p}}_{0/\infty}(\mathsf{A}_h))$, i. e., $(w, v) \in \mathsf{CUL}(v^{\overline{p}}_{0/\infty}(\mathsf{A}_h))$.
– $(w, v) \in \mathsf{CUL}(v^{\overline{p}}_{0/\infty}(\mathsf{A}_h))$. Then, as before, w is accepted by $v^{\overline{p}}_{0/\infty}(v(\mathsf{A}_h))$ and $v(\mathsf{A}_h)$ is an L/U-PTA. Then using the bound of [HRSV02, Proposition 4.4], which we will call $\mathcal{N}$ here, we know that if v' is the valuation of hidden parameters that assigns 0 to lower bound parameters and $\mathcal{N}$ to upper bound parameters then w is accepted by $v'(v(\mathsf{A}_h))$, which in turn means that $(w, v) \in \mathsf{CUL}(\mathsf{A}_h)$.

$\supseteq$ The converse is trivial: let A be an L/U-PTA $\mathsf{A} = (\Sigma, L, l_0, F, X, P, I, E)$. Then A is in itself an hL/U-PTA with no hidden parameters, and therefore there exists an hL/U-PTA (itself) with the same constrained untimed language as A.

□