Execution-Time Opacity Problems in One-Clock Parametric Timed Automata

Étienne André
Université Sorbonne Paris Nord, LIPN, CNRS UMR 7030, F-93430 Villetaneuse, France
Institut Universitaire de France (IUF), Paris, France

Johan Arcile
IBISC, Univ Evry, Université Paris-Saclay, 91025 Evry, France

Engel Lefaucheux
Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

Abstract

Parametric timed automata (PTAs) extend the concept of timed automata, by allowing timing delays not only specified by concrete values but also by parameters, allowing the analysis of systems with uncertainty regarding timing behaviors. The full execution-time opacity is defined as the problem in which an attacker must never be able to deduce whether some private location was visited, by only observing the execution time. The problem of full ET-opacity emptiness (i.e., the emptiness over the parameter valuations for which full execution-time opacity is satisfied) is known to be undecidable for general PTAs. We therefore focus here on one-clock PTAs with integer-valued parameters over dense time. We show that the full ET-opacity emptiness is undecidable for a sufficiently large number of parameters, but is decidable for a single parameter, and exact synthesis can be effectively achieved. Our proofs rely on a novel construction as well as on variants of Presburger arithmetics. We finally prove an additional decidability result on an existential variant of execution-time opacity.

2012 ACM Subject Classification Theory of computation → Timed and hybrid models; Security and privacy → Logic and verification

Keywords and phrases Timed opacity, Parametric timed automata, Presburger arithmetic

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2024.3

Related Version *Full Version*: https://arxiv.org/abs/2410.01659

Funding This work is partially supported by the ANR-NRF French-Singaporean research program ProMiS (ANR-19-CE25-0015 / 2019 ANR NRF 0092) and by ANR BisoUS (ANR-22-CE48-0012).

1 Introduction

As surveyed in [13], for some systems, private information may be deduced simply by observation of public information. For example, it may be possible to infer the content of some memory space from the access times of a cryptographic module.

The notion of *opacity* [28, 15] concerns information leaks from a system to an attacker; that is, it expresses the power of the attacker to deduce some secret information based on some publicly observable behaviors. If an attacker observing a subset of the actions cannot deduce whether a given sequence of actions has been performed, then the system is opaque. Time particularly influences the deductive capabilities of the attacker. It has been shown in [19] that it is possible for models that are opaque when timing constraints are omitted, to be non-opaque when those constraints are added to the models.

For this reason, the notion is extended to *timed* opacity in [17], where the attacker can also observe time. The input model is timed automata (TAs) [1], a formalism extending finite-state automata with real-time variables called *clocks*. It is proved in [17] that this version of timed opacity is undecidable for TAs.

In [9], a less powerful version of opacity is proposed, where the attacker has access only to the system execution time and aims at deducing whether a private location was visited during the system execution. This version of timed opacity is called *execution-time opacity (ET-opacity)*. Two main problems are considered in [9]: 1) the existence of at least one execution time for which the system is ET-opaque (∃-*ET-opacity*), and 2) whether *all* execution times are such that the system is ET-opaque (called *full ET-opacity*). These two notions of opacity are proved to be decidable for TAs [7]. In the same works, the authors then extend ET-opacity to parametric timed automata (PTAs) [2]. PTAs are an extension of TAs where timed constraints can be expressed with timing parameters instead of integer constants, allowing to model uncertainty or lack of knowledge. The two problems come with two flavors: 1) *emptiness* problems: whether the set of parameter valuations guaranteeing a given version of opacity (∃-ET-opacity or full ET-opacity) is empty or not, and 2) *synthesis* problems: synthesize all parameter valuations for which a given version of opacity holds. Both emptiness problems ∃`OE` (∃-ET-opacity emptiness) and `FOE` (full-ET-opacity emptiness) have been shown to be undecidable for PTAs, while decidable subclasses are exhibited [9, 7]. A semi-algorithm (i.e., that may not terminate, but is correct if it does) is provided to solve ∃-ET-opacity synthesis (hereafter ∃`OS`) in [9].

1.1 Contributions

We address here full-ET-opacity emptiness (`FOE`) and synthesis (`FOS`), and ∃-ET-opacity emptiness (∃`OE`) and synthesis (∃`OS`), for PTAs with integer-valued parameters over dense time with the following main theoretical contributions:

1. We prove that **`FOE` is undecidable** (Corollary 27) for PTAs with a single clock and a sufficiently large number of parameters.
2. We prove in contrast that **`FOE` is decidable** (Corollary 28) for PTAs with a single clock and a single parameter.
3. We prove that **∃`OE` is decidable** (Theorem 29) for PTAs with a single clock and arbitrarily many parameters. We also exhibit a better complexity for a single parameter over discrete time (Theorem 31).

We focus on one-clock PTAs, as virtually all problems are undecidable for 3 clocks [5], and the 2-clock case is an extremely difficult problem, already for reachability [21]. Our contributions are summarized in Table 1. In order to prove these results, we improve on the semi-algorithm from [9] for ∃`OS` and provide one for `FOS`. These solutions are based on the novel notion of *parametric execution times* (PET). The PET of a PTA is the total elapsed time and associated parameter valuations on all paths between two given locations. We provide a semi-algorithm for the computation of PET, that builds upon reachability synthesis (i.e., the synthesis of parameter valuations for which a set of locations are reachable) for which a semi-algorithm already exists [24]. We then show how to resolve ∃`OS` and `FOS` problems by performing set operations on PET of two complementary subsets of the PTA where we respectively consider only private paths and only non-private paths.

We then solve the full ET-opacity emptiness (`FOE`) problem for PTAs with 1 clock and 1 parameter, by rewriting the problems in a parametric variant of Presburger arithmetic. This is done by 1) providing a sound and complete method for encoding infinite PET for PTAs with 1 clock and arbitrarily many parameters over dense time; and 2) translating them into parametric semi-linear sets, a formalism defined and studied in [27]. With these ingredients, we notably prove that: 1) `FOE` is undecidable in general for PTAs with 1 clock and sufficiently many parameters. This is done by reducing a known undecidable problem of parametric Presburger arithmetic (whose undecidability comes from Hilbert's 10th problem)

to the FOE problem in this context. 2) ∃OE is decidable for PTAs with 1 clock and arbitrarily many parameters. This is done by reducing ∃OE to the existential fragment of Presburger arithmetic with divisibility, known to be decidable.

1.2 Related works

The undecidability of timed opacity proved in [17] leaves hope for decidability only by modifying the problem (as in [9, 7]), or by restraining the model. In [31, 32], (initial state) opacity is shown to be decidable on a restricted subclass of TAs called real-time automata [18]. In [3], a notion of *timed bounded opacity*, where the secret has an expiration date, and over a time-bounded framework, is proved decidable. Opacity over subclasses of TAs (such as one-clock or one-actions TAs) is considered in [6, 4] and over discrete time in [25].

In [9], ∃-ET-opacity synthesis (∃OS) is solved using a semi-algorithm. The method is based on a self-composition of the PTA with m parameters and n clocks, where the resulting model consists of $m+1$ parameters and $2n+1$ clocks. The method terminates if the symbolic state space of this self-composition is finite. Our work proposes in contrast an approach based on set operations on parametric execution times (PET) of both complementary subsets of the PTA where we respectively consider only private paths and only non-private paths. Those submodels are each composed of $m+1$ parameters and $n+1$ clocks. Our new method terminates if the symbolic state spaces of both submodels are finite. Another improvement is that the method described here also supports full timed opacity synthesis (FOS).

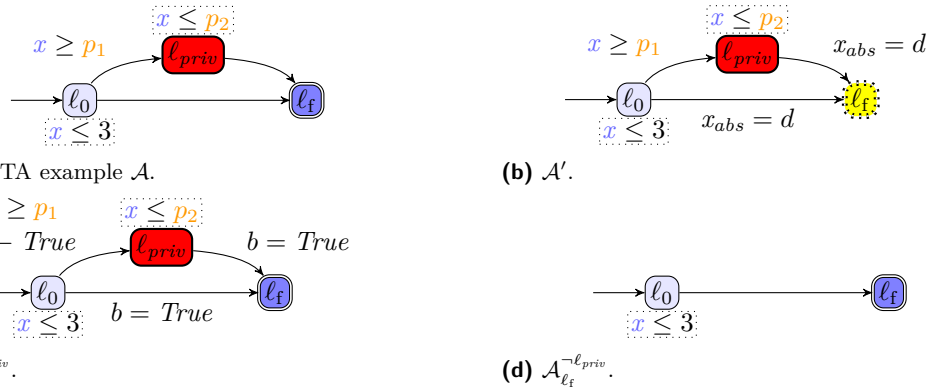
The reachability emptiness problem (i.e., the emptiness over the valuations set for which a given target location is reachable) is known to be undecidable in general since [2]. The rare decidable settings require a look at the number of parametric clocks (i.e., compared at least once in a guard or invariant to a parameter), non-parametric clocks and parameters; throughout this paper, we denote these 3 numbers using a triple (pc, npc, p). Reachability emptiness is decidable for $(1, *, *)$-PTAs ("$*$" denotes "arbitrarily many" for decidable cases, and "sufficiently many" for undecidable cases) over discrete time [2] or dense time with integer-valued parameters [12], for $(1, 0, *)$-PTAs over dense time over rational-valued parameters [10], and for $(2, *, 1)$-PTAs over discrete time [16, 21]; and it is undecidable for $(3, *, 1)$-PTAs over discrete or dense time [12], and for $(1, 3, 1)$-PTAs over dense time only for rational-valued parameters [29]. See [5] for a complete survey as of 2019.

Section 2 recalls the necessary preliminaries. Section 3 introduces one of our main technical proof ingredients, i.e., the definition of PET, and PET-based semi-algorithms for ∃OS and FOS. Section 4 considers the FOE problem over $(1, 0, *)$-PTAs (undecidable) and $(1, 0, 1)$-PTAs (decidable). Section 5 proves decidability of ∃OE for $(1, 0, *)$-PTAs. We also give a better complexity for $(1, 0, 1)$-PTAs over discrete time. Section 6 concludes.

2 Preliminaries

We let $\mathbb{T}$ be the domain of the time, which will be either non-negative reals $\mathbb{R}_{\geq 0}$ (continuous-time semantics) or naturals $\mathbb{N}$ (discrete-time semantics). Unless otherwise specified, we assume $\mathbb{T} = \mathbb{R}_{\geq 0}$.

Clocks are real-valued variables that all evolve over time at the same rate. We assume a set $\mathbb{X} = \{x_1, \ldots, x_H\}$ of *clocks*. A *clock valuation* is a function $\mu : \mathbb{X} \to \mathbb{T}$. We write $\vec{0}$ for the clock valuation assigning 0 to all clocks. Given a constant $\gamma \in \mathbb{T}$, $\mu + \gamma$ denotes the valuation s.t. $(\mu + \gamma)(x) = \mu(x) + \gamma$, for all $x \in \mathbb{X}$. Given $R \subseteq \mathbb{X}$, 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.



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

(b) $\mathcal{A}'$.

(c) $\mathcal{A}^{\ell_{priv}}_{\ell_f}$.

(d) $\mathcal{A}^{\neg\ell_{priv}}_{\ell_f}$.

Figure 1 A PTA example and its transformed versions. The yellow dotted location is urgent.

A *(timing) parameter* is an unknown integer-valued constant of a model. We assume a set $\mathbb{P} = \{p_1, \ldots, p_M\}$ of *parameters*. A *parameter valuation* v is a function $v : \mathbb{P} \to \mathbb{N}$.

We assume $\bowtie \in \{<, \leq, =, \geq, >\}$. A *clock guard* C is a conjunction of inequalities over $\mathbb{X} \cup \mathbb{P}$ of the form $x \bowtie \sum_{1 \leq i \leq M} \alpha_i p_i + \gamma$, with $x \in \mathbb{X}$, $p_i \in \mathbb{P}$, and $\alpha_i, \gamma \in \mathbb{Z}$. Given C, we write $\mu \models v(C)$ if the expression obtained by replacing each x with $\mu(x)$ and each p with $v(p)$ in C evaluates to true.

2.1 Parametric timed automata

Parametric timed automata (PTAs) extend TAs with parameters within guards and invariants in place of integer constants [2]. We also add to the standard definition of PTAs a special private location, which will be used to define our subsequent opacity concepts.

▶ **Definition 1** (PTA [2]). *A parametric timed automaton (PTA) [2] $\mathcal{A}$ is a tuple $\mathcal{A} = (\Sigma, L, \ell_0, \ell_{priv}, \ell_f, \mathbb{X}, \mathbb{P}, I, E)$, where: 1) Σ is a finite set of actions; 2) L is a finite set of locations; 3) $\ell_0 \in L$ is the initial location; 4) $\ell_{priv} \in L$ is a special private location; 5) $\ell_f \in L$ is the final location; 6) $\mathbb{X}$ is a finite set of clocks; 7) $\mathbb{P}$ is a finite set of parameters; 8) I is the invariant, assigning to every $\ell \in L$ a clock guard $I(\ell)$ (called* invariant*); 9) E is a finite set of edges $e = (\ell, g, a, R, \ell')$ where $\ell, \ell' \in L$ are the source and target locations, $a \in \Sigma$, $R \subseteq \mathbb{X}$ is a set of clocks to be reset, and g is a clock guard.*

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

▶ **Definition 2** (Reset-free PTA). *A* reset-free PTA $\mathcal{A} = (\Sigma, L, \ell_0, \ell_{priv}, \ell_f, \mathbb{X}, \mathbb{P}, I, E)$ *is a PTA where* $\forall\ (\ell, g, a, R, \ell') \in E,\ R = \emptyset$.

▶ **Example 3.** Consider the PTA $\mathcal{A}$ in Figure 1a. It has three locations, one clock and two parameters (actions are omitted). "$x \leq p_2$" is the invariant of ℓ_{priv}, and the transition from ℓ_0 to ℓ_{priv} has guard "$x \geq p_1$". In this example, x is never reset, and therefore $\mathcal{A}$ happens to be reset-free.

▶ **Definition 4** (Semantics of a timed automaton (TA) [1]). *Given a PTA $\mathcal{A} = (\Sigma, L, \ell_0, \ell_{priv}, \ell_f, \mathbb{X}, \mathbb{P}, I, E)$ and a parameter valuation v, the semantics of the TA $v(\mathcal{A})$ is given by the timed transition system (TTS) [22] $\mathfrak{T}_{v(\mathcal{A})} = (\mathfrak{S}, \mathfrak{s}_0, \Sigma \cup \mathbb{R}_{\geq 0}, \to)$, with*

1. $\mathfrak{S} = \{(\ell, \mu) \in L \times \mathbb{R}^H_{\geq 0} \mid \mu \models v(I(\ell))\}$, $\mathfrak{s}_0 = (\ell_0, \vec{0})$,
2. $\to$ *consists of the discrete and (continuous) delay transition relations:*

a. *discrete transitions:* $(\ell,\mu) \overset{e}{\mapsto} (\ell',\mu')$, *if* $(\ell,\mu),(\ell',\mu') \in \mathfrak{S}$, *and there exists* $e = (\ell,g,a,R,\ell') \in E$, *such that* $\mu' = [\mu]_R$, *and* $\mu \models v(g)$.
b. *delay transitions:* $(\ell,\mu) \overset{\gamma}{\mapsto} (\ell,\mu+\gamma)$, *with* $\gamma \in \mathbb{R}_{\geq 0}$, *if* $\forall \gamma' \in [0,\gamma], (\ell,\mu+\gamma') \in \mathfrak{S}$.

Moreover we write $(\ell,\mu) \overset{(\gamma,e)}{\longrightarrow} (\ell',\mu')$ for a combination of a delay and discrete transition if $\exists \mu'' : (\ell,\mu) \overset{\gamma}{\mapsto} (\ell,\mu'') \overset{e}{\mapsto} (\ell',\mu')$.

Given a TA $v(\mathcal{A})$ with concrete semantics $(\mathfrak{S},\mathfrak{s}_0,\Sigma \cup \mathbb{R}_{\geq 0},\to)$, we refer to the states of $\mathfrak{S}$ as the *concrete states* of $v(\mathcal{A})$. A *run* of $v(\mathcal{A})$ is an alternating sequence of concrete states of $v(\mathcal{A})$ and pairs of edges and delays starting from the initial state $\mathfrak{s}_0$ of the form $(\ell_0,\mu_0),(d_0,e_0),(\ell_1,\mu_1),\cdots$ with $i = 0,1,\ldots$, $e_i \in E$, $d_i \in \mathbb{R}_{\geq 0}$ and $(\ell_i,\mu_i) \overset{(d_i,e_i)}{\longrightarrow} (\ell_{i+1},\mu_{i+1})$.

Given a state $\mathfrak{s} = (\ell,\mu)$, we say that $\mathfrak{s}$ is *reachable* in $v(\mathcal{A})$ if $\mathfrak{s}$ appears in a run of $v(\mathcal{A})$. By extension, we say that ℓ is reachable in $v(\mathcal{A})$; and by extension again, given a set L_{target} of locations, we say that L_{target} is reachable in $v(\mathcal{A})$ if there exists $\ell \in L_{target}$ such that ℓ is reachable in $v(\mathcal{A})$.

Given a finite run $\rho : (\ell_0,\mu_0),(d_0,e_0),(\ell_1,\mu_1),\cdots,(d_{i-1},e_{i-1}),(\ell_n,\mu_n)$, the *duration* of ρ is $dur(\rho) = \sum_{0\leq i\leq n-1} d_i$. We also say that ℓ_n is reachable in time $dur(\rho)$.

Let us now recall the symbolic semantics of PTAs (see e.g., [23]). We first define operations on constraints. A *linear term* over $\mathbb{X} \cup \mathbb{P}$ is of the form $\sum_{1\leq i\leq H} \alpha_i x_i + \sum_{1\leq j\leq M} \beta_j p_j + \gamma$, with $x_i \in \mathbb{X}$, $p_j \in \mathbb{P}$, and $\alpha_i,\beta_j,\gamma \in \mathbb{Z}$. A *constraint* $\mathbf{C}$ (i.e., a convex polyhedron) over $\mathbb{X} \cup \mathbb{P}$ is a conjunction of inequalities of the form $lt \bowtie 0$, where lt is a linear term. Given a parameter valuation v, $v(\mathbf{C})$ denotes the constraint over $\mathbb{X}$ obtained by replacing each parameter p in $\mathbf{C}$ with $v(p)$. Likewise, given a clock valuation μ, $\mu(v(\mathbf{C}))$ denotes the expression obtained by replacing each clock x in $v(\mathbf{C})$ with $\mu(x)$. We write $\mu \models v(\mathbf{C})$ whenever $\mu(v(\mathbf{C}))$ evaluates to true. We say that v *satisfies* $\mathbf{C}$, denoted by $v \models \mathbf{C}$, if the set of clock valuations satisfying $v(\mathbf{C})$ is nonempty. We say that $\mathbf{C}$ is *satisfiable* if $\exists \mu, v$ s.t. $\mu \models v(\mathbf{C})$. We define the *time elapsing* of $\mathbf{C}$, denoted by $\mathbf{C}^{\nearrow}$, as the constraint over $\mathbb{X}$ and $\mathbb{P}$ obtained from $\mathbf{C}$ by delaying all clocks by an arbitrary amount of time. That is, $\mu' \models v(\mathbf{C}^{\nearrow})$ if $\exists \mu : \mathbb{X} \to \mathbb{R}_{\geq 0}, \exists \gamma \in \mathbb{R}_{\geq 0}$ s.t. $\mu \models v(\mathbf{C}) \wedge \mu' = \mu + \gamma$. Given $R \subseteq \mathbb{X}$, we define the *reset* of $\mathbf{C}$, denoted by $[\mathbf{C}]_R$, as the constraint obtained from $\mathbf{C}$ by resetting the clocks in R to 0, and keeping the other clocks unchanged. That is,

$$\mu' \models v([\mathbf{C}]_R) \text{ if } \exists \mu : \mathbb{X} \to \mathbb{R}_{\geq 0} \text{ s.t. } \mu \models v(\mathbf{C}) \wedge \forall x \in \mathbb{X} \begin{cases} \mu'(x) = 0 & \text{if } x \in R \\ \mu'(x) = \mu(x) & \text{otherwise.} \end{cases}$$

We denote by $\mathbf{C}{\downarrow_{\mathbb{P}}}$ the projection of $\mathbf{C}$ onto $\mathbb{P}$, i.e., obtained by eliminating the variables not in $\mathbb{P}$ (e.g., using Fourier-Motzkin [30]).

▶ **Definition 5** (Symbolic state). *A symbolic state is a pair* $(\ell,\mathbf{C})$ *where* $\ell \in L$ *is a location, and* $\mathbf{C}$ *its associated parametric zone.*

▶ **Definition 6** (Symbolic semantics). *Given a PTA* $\mathcal{A} = (\Sigma,L,\ell_0,\ell_{priv},\ell_f,\mathbb{X},\mathbb{P},I,E)$, *the symbolic semantics of* $\mathcal{A}$ *is the labeled transition system called* parametric zone graph $\mathbf{PZG}(\mathcal{A}) = (E,\mathbf{S},\mathbf{s}_0,\Rightarrow)$, *with*

- $\mathbf{S} = \{(\ell,\mathbf{C}) \mid \mathbf{C} \subseteq I(\ell)\}$, $\mathbf{s}_0 = \left(\ell_0, \left(\bigwedge_{1\leq i\leq H} x_i = 0\right)^{\nearrow} \wedge I(\ell_0)\right)$, *and*
- $\left((\ell,\mathbf{C}),e,(\ell',\mathbf{C}')\right) \in \Rightarrow$ *if* $e = (\ell,g,a,R,\ell') \in E$ *and*

$$\mathbf{C}' = \left([(\mathbf{C} \wedge g)]_R \wedge I(\ell')\right)^{\nearrow} \wedge I(\ell') \text{ with } \mathbf{C}' \text{ satisfiable.}$$

That is, in the parametric zone graph, nodes are symbolic states, and arcs are labeled by *edges* of the original PTA.

2.2 Reachability synthesis

We use reachability synthesis to solve the problems defined in Section 2.3. This procedure, called EFsynth, takes as input a PTA $\mathcal{A}$ and a set of target locations L_{target}, and attempts to synthesize all parameter valuations v for which L_{target} is reachable in $v(\mathcal{A})$. $\mathsf{EFsynth}(\mathcal{A}, L_{target})$ was formalized in e.g., [24] and is a procedure that may not terminate, but that computes an exact result (sound and complete) if it terminates.

2.3 Execution-time opacity problems

We recall here the notion of execution-time opacity [9, 7]. This form of opacity is such that the observation is limited to the time to reach a given location. This section recalls relevant definitions from [9, 7].

Given a TA $v(\mathcal{A})$ and a run ρ, we say that ℓ_{priv} is *visited on the way to* ℓ_f *in* ρ if ρ is of the form $(\ell_0, \mu_0), (d_0, e_0), (\ell_1, \mu_1), \cdots, (\ell_\mathrm{m}, \mu_m), (d_m, e_m), \cdots (\ell_\mathrm{n}, \mu_n)$
for some $m, n \in \mathbb{N}$ such that $\ell_\mathrm{m} = \ell_{priv}$, $\ell_\mathrm{n} = \ell_\mathrm{f}$ and $\forall 0 \leq i \leq n-1, \ell_\mathrm{i} \neq \ell_\mathrm{f}$. We denote by $\mathit{Visit}^{priv}(v(\mathcal{A}))$ the set of those runs, and refer to them as *private* runs. We denote by $\mathit{DVisit}^{priv}(v(\mathcal{A}))$ the set of all the durations of these runs.

Conversely, we say that ℓ_{priv} is *avoided on the way to* ℓ_f *in* ρ if ρ is of the form $(\ell_0, \mu_0), (d_0, e_0), (\ell_1, \mu_1), \cdots, (\ell_\mathrm{n}, \mu_n)$
with $\ell_\mathrm{n} = \ell_\mathrm{f}$ and $\forall 0 \leq i < n, \ell_\mathrm{i} \notin \{\ell_{priv}, \ell_\mathrm{f}\}$. We denote the set of those runs by $\mathit{Visit}^{\overline{priv}}(v(\mathcal{A}))$, referring to them as *public* runs, and by $\mathit{DVisit}^{\overline{priv}}(v(\mathcal{A}))$ the set of all the durations of these public runs. Therefore, $\mathit{DVisit}^{priv}(v(\mathcal{A}))$ (resp. $\mathit{DVisit}^{\overline{priv}}(v(\mathcal{A}))$) is the set of all the durations of the runs for which ℓ_{priv} is visited (resp. avoided) on the way to ℓ_f. These concepts can be seen as the set of execution times from the initial location ℓ_0 to the final location ℓ_f while visiting (resp. not visiting) a private location ℓ_{priv}. Observe that, from the definition of the duration of a run, this "execution time" does not include the time spent in ℓ_f.

We now recall formally the concept of "execution-time opacity (ET-opacity) for a set of durations (or execution times) D": a system is *ET-opaque for execution times* D whenever, for any duration in D, it is not possible to deduce whether the system visited ℓ_{priv} or not.

▶ **Definition 7** (Execution-time opacity (ET-opacity) for D)**.** *Given a TA* $v(\mathcal{A})$ *and a set of execution times* D*, we say that* $v(\mathcal{A})$ *is* execution-time opaque (ET-opaque) for execution times D *if* $D \subseteq (\mathit{DVisit}^{priv}(v(\mathcal{A})) \cap \mathit{DVisit}^{\overline{priv}}(v(\mathcal{A})))$*.*

In the following, we will be interested in the *existence* of such an execution time. We say that a TA is ∃-ET-opaque if it is ET-opaque for a non-empty set of execution times.

▶ **Definition 8** (∃-ET-opacity)**.** *A TA* $v(\mathcal{A})$ *is* ∃-ET-opaque *if* $(\mathit{DVisit}^{priv}(v(\mathcal{A})) \cap \mathit{DVisit}^{\overline{priv}}(v(\mathcal{A}))) \neq \emptyset$*.*

In addition, a system is *fully ET-opaque* if, for any possible measured execution time, an attacker is not able to deduce whether ℓ_{priv} was visited or not.

▶ **Definition 9** (full ET-opacity)**.** *A TA* $v(\mathcal{A})$ *is* fully ET-opaque *if* $\mathit{DVisit}^{priv}(v(\mathcal{A})) = \mathit{DVisit}^{\overline{priv}}(v(\mathcal{A}))$*.*

▶ **Example 10.** Consider again the PTA $\mathcal{A}$ in Figure 1a. Let v s.t. $v(p_1) = 1$ and $v(p_2) = 4$. Then $v(\mathcal{A})$ is ∃-ET-opaque since there is at least one execution time for which $v(\mathcal{A})$ is ET-opaque. Here, $v(\mathcal{A})$ is ET-opaque for execution times $[1, 3]$. However, $v(\mathcal{A})$ is not fully

ET-opaque since there is at least one execution time for which $v(\mathcal{A})$ is not ET-opaque. Here, $v(\mathcal{A})$ is not ET-opaque for execution times $[0, 1)$ (which can only occur on a public run) and for execution times $(3, 4]$ (which can only occur on a private run).

Let us consider the following decision problems:

∃-ET-opacity p emptiness problem (∃OE):
INPUT: A PTA $\mathcal{A}$
PROBLEM: Decide the emptiness of the set of valuations v s.t. $v(\mathcal{A})$ is ∃-ET-opaque.

Full ET-opacity p emptiness problem (FOE):
INPUT: A PTA $\mathcal{A}$
PROBLEM: Decide the emptiness of the set of valuations v s.t. $v(\mathcal{A})$ is fully ET-opaque.

The synthesis counterpart allows for a higher-level problem aiming at synthesizing (ideally the entire set of) parameter valuations v for which $v(\mathcal{A})$ is ∃-ET-opaque or fully ET-opaque.

∃-ET-opacity p synthesis problem (∃OS):
INPUT: A PTA $\mathcal{A}$
PROBLEM: Synthesize the set of all valuations v s.t. $v(\mathcal{A})$ is ∃-ET-opaque.

Full ET-opacity p synthesis problem (FOS):
INPUT: A PTA $\mathcal{A}$
PROBLEM: Synthesize the set of all valuations v s.t. $v(\mathcal{A})$ is fully ET-opaque.

3 A parametric execution times-based semi-algorithm for ∃OS and FOS

One of our main results is the proof that both ∃OS and FOS can be deduced from set operations on two sets representing respectively all the durations and parameter valuations of the runs for which ℓ_{priv} is reached (resp. avoided) on the way to ℓ_f. Those sets can be seen as a parametrized version of $DVisit^{priv}(v(\mathcal{A}))$ and $\overline{DVisit^{priv}}(v(\mathcal{A}))$. In order to compute such sets, we propose here the novel notion of parametric execution times. (Note that our partial solution for PET construction and semi-algorithms for ∃OS and FOS work perfectly for *rational*-valued parameters too, and that they are not restricted to 1-clock PTAs.)

3.1 Parametric execution times

The parametric execution times (PET) are the parameter valuations and execution times of the runs to ℓ_f.

▶ **Definition 11.** *Given a PTA $\mathcal{A}$ with final location ℓ_f, the* parametric execution times *of $\mathcal{A}$ are defined as $PET(\mathcal{A}) = \{(v, d) \mid \exists \rho$ in $v(\mathcal{A})$ such that $d = dur(\rho) \wedge \rho$ is of the form $(\ell_0, \mu_0), (d_0, e_0), \cdots, (\ell_n, \mu_n)$ for some $n \in \mathbb{N}$ such that $\ell_n = \ell_f$ and $\forall 0 \leq i \leq n-1, \ell_i \neq \ell_f\}$.*

By definition, we only consider paths up to the point where ℓ_f is reached, meaning that execution times do not include the time elapsed in ℓ_f, and that runs that reach ℓ_f more than once are only considered up to their first visit of ℓ_f.

▶ **Example 12.** Consider again the PTA $\mathcal{A}$ in Figure 1a. Then $PET(\mathcal{A})$ is $(d \leq 3 \wedge p_1 \geq 0 \wedge p_2 \geq 0) \vee (0 \leq p_1 \leq 3 \wedge p_1 \leq d \leq p_2)$.

Partial solution

Synthesizing parametric execution times is in fact equivalent to a reachability synthesis where the PTA is enriched (in particular by adding a clock measuring the total execution time).

▶ **Proposition 13.** *Let $\mathcal{A}$ be a PTA, and ℓ_{f} the final location of $\mathcal{A}$.*
Let $\mathcal{A}'$ be a copy of $\mathcal{A}$ s.t.:
- *a clock x_{abs} is added and initialized at 0 (it does not occur in any guard or reset);*
- *a parameter d is added;*
- *ℓ_{f} is made* urgent *(i.e., time is not allowed to pass in ℓ_{f}), all outgoing edges from ℓ_{f} are pruned and a guard $x_{abs} = d$ is added to all incoming edges to ℓ_{f}.*

Then, $PET(\mathcal{A}) = \mathsf{EFsynth}(\mathcal{A}', \{\ell_{\mathrm{f}}\})$.

Proof. By having ℓ_{f} being urgent and removing its outgoing edges, we ensure that the runs that reach ℓ_{f} in $\mathcal{A}'$ are all of the form $(\ell_0, \mu_0), (d_0, e_0), \cdots, (\ell_n, \mu_n)$ for some $n \in \mathbb{N}$ such that $\ell_n = \ell'$ and $\forall 0 \leq i \leq n-1, \ell_i \neq \ell'$. By having a clock x_{abs} that is never reset and ℓ_{f} being urgent, we ensure that for any run ρ that reaches ℓ_{f} in $\mathcal{A}'$, the value of x_{abs} in the final state if equals to $dur(\rho)$. By having a guard $x_{abs} = d$ on all incoming edges to ℓ_{f}, we ensure that $d = dur(\rho)$ on any run ρ that reaches ℓ_{f}.

Therefore, $\mathsf{EFsynth}(\mathcal{A}', \{\ell_{\mathrm{f}}\})$ contains all parameter valuations of the runs to ℓ_{f} in $\mathcal{A}$ that stop once ℓ_{f} is reached, along with the duration of those runs contained in d. ◀

▶ **Example 14.** Consider again the PTA $\mathcal{A}$ in Figure 1a. Then $\mathcal{A}'$ is given in Figure 1b.

As per Lemma 33 in Section A, there exist semi-algorithms for reachability synthesis, and hence for the PET synthesis problem – although they do not guarantee termination.

3.2 ∃OS and FOS problems

Now, we detail how the PET can be used to compute the solution to both ∃OS and FOS. To do so, we will go through a (larger) intermediate problem: the synthesis of both parameter valuations v *and* execution times for which $v(\mathcal{A})$ is ET-opaque.

∃-ET-opacity p-d synthesis problem (d-∃OS):
INPUT: A PTA $\mathcal{A}$
PROBLEM: Synthesize the set of parameter valuations v and execution times d s.t. $v(\mathcal{A})$ is ∃-ET-opaque and $v(\mathcal{A})$ is ET-opaque for execution time d.

Full ET-opacity p-d synthesis problem (d-FOS):
INPUT: A PTA $\mathcal{A}$
PROBLEM: Synthesize the set of parameter valuations v and execution times d s.t. $v(\mathcal{A})$ is fully ET-opaque and d is the set of durations of all runs in $v(\mathcal{A})$.

First, given a PTA $\mathcal{A}$ and two locations ℓ_{f} and ℓ_{priv} of $\mathcal{A}$, let us formally define both sets representing respectively all the durations and parameter valuations of the runs for which ℓ_{priv} is reached (resp. avoided) on the way to ℓ_{f}.

Let $\mathcal{A}_{\ell_{\mathrm{f}}}^{\ell_{priv}}$ be a copy of $\mathcal{A}$ s.t.: 1) a Boolean variable[1] b is added and initialized to *False*, 2) b is set to *True* on all incoming edges to ℓ_{priv}, 3) a guard $b =$ *True* is added to all incoming edges to ℓ_{f}. The PTA $\mathcal{A}_{\ell_{\mathrm{f}}}^{\ell_{priv}}$ contains all runs of $\mathcal{A}$ for which ℓ_{priv} is reached on the way to ℓ_{f}, and $PET(\mathcal{A}_{\ell_{\mathrm{f}}}^{\ell_{priv}})$ contains the durations and parameter valuations of those runs.

[1] Which is a convenient syntactic sugar for doubling the number of locations.

Let $\mathcal{A}_{\ell_f}^{\neg \ell_{priv}}$ be a copy of $\mathcal{A}$ s.t. all incoming and outgoing edges to and from ℓ_{priv} are pruned. The PTA $\mathcal{A}_{\ell_f}^{\neg \ell_{priv}}$ contains all runs of $\mathcal{A}$ for which ℓ_{priv} is avoided on the way to ℓ_f, and $PET(\mathcal{A}_{\ell_f}^{\neg \ell_{priv}})$ contains the durations and parameter valuations of those runs.

▶ **Example 15.** Consider again the PTA $\mathcal{A}$ in Figure 1a. Then $\mathcal{A}_{\ell_f}^{\ell_{priv}}$ is given in Figure 1c, and $\mathcal{A}_{\ell_f}^{\neg \ell_{priv}}$ is given in Figure 1d.

▶ **Proposition 16.** *Given a PTA $\mathcal{A}$, we have:* $\texttt{d-}\exists\texttt{OS}(\mathcal{A}) = PET(\mathcal{A}_{\ell_f}^{\ell_{priv}}) \cap PET(\mathcal{A}_{\ell_f}^{\neg \ell_{priv}})$.

Proof. By definition, $\texttt{d-}\exists\texttt{OS}(\mathcal{A})$ is the synthesis of parameter valuations v and execution times D_v such that $v(\mathcal{A})$ is opaque w.r.t. ℓ_{priv} on the way to ℓ_f for these execution times D_v. This means that $\texttt{d-}\exists\texttt{OS}(\mathcal{A})$ contains exactly all parameter valuations and execution times for which there exist both at least one run in $\mathcal{A}_{\ell_f}^{\ell_{priv}}$ and at least one run in $\mathcal{A}_{\ell_f}^{\neg \ell_{priv}}$. Since PET are the synthesis of the parameter valuations and execution times up to the final location, $\texttt{d-}\exists\texttt{OS}(\mathcal{A})$ is equivalent to the intersection of the $PET(\mathcal{A}_{\ell_f}^{\ell_{priv}})$ and $PET(\mathcal{A}_{\ell_f}^{\neg \ell_{priv}})$. ◀

▶ **Example 17.** Consider again the PTA $\mathcal{A}$ in Figure 1a. Then $PET(\mathcal{A}_{\ell_f}^{\ell_{priv}})$ is $p_1 \leq d \leq p_2 \wedge 0 \leq p_1 \leq 3$. Moreover, $PET(\mathcal{A}_{\ell_f}^{\neg \ell_{priv}})$ is $0 \leq d \leq 3 \wedge p_1 \geq 0 \wedge p_2 \geq 0$. Hence, $\texttt{d-}\exists\texttt{OS}(\mathcal{A})$ is $0 \leq p_1 \leq d \leq p_2 \wedge d \leq 3$.

In order to compute $\texttt{d-FOS}(\mathcal{A})$, we need to remove from $\texttt{d-}\exists\texttt{OS}(\mathcal{A})$ all parameter valuations v s.t. there is at least one run to ℓ_f in $v(\mathcal{A})$ whose duration is not in the set of execution times for which $v(\mathcal{A})$ is ET-opaque. Parameter valuations and durations of such runs are included in $PET(\mathcal{A}) \setminus \texttt{d-}\exists\texttt{OS}(\mathcal{A})$, which is also the difference between $PET(\mathcal{A}_{\ell_f}^{\ell_{priv}})$ and $PET(\mathcal{A}_{\ell_f}^{\neg \ell_{priv}})$. We note that difference as

$$\mathit{Diff}(\mathcal{A}) = \left(PET(\mathcal{A}_{\ell_f}^{\ell_{priv}}) \cup PET(\mathcal{A}_{\ell_f}^{\neg \ell_{priv}})\right) \setminus \left(PET(\mathcal{A}_{\ell_f}^{\ell_{priv}}) \cap PET(\mathcal{A}_{\ell_f}^{\neg \ell_{priv}})\right)$$

$\mathit{Diff}(\mathcal{A})$ is made of a union of convex polyhedra $\mathbf{C}$ over $\mathbb{P}$ (i.e., the parameters of $\mathcal{A}$) and d, which is the duration of runs. The parameter values in those polyhedra are the ones we do not want to see in $\texttt{d-FOS}(\mathcal{A})$. Our solution thus consists in removing from $\texttt{d-}\exists\texttt{OS}(\mathcal{A})$ the values of $\mathbb{P}$ in $\mathit{Diff}(\mathcal{A})$.

▶ **Proposition 18.** *Given a PTA $\mathcal{A}$ with parameter set $\mathbb{P}$:* $\texttt{d-FOS}(\mathcal{A}) = \texttt{d-}\exists\texttt{OS}(\mathcal{A}) \setminus \mathit{Diff}(\mathcal{A})\downarrow_{\mathbb{P}}$.

Proof. See Section B.1. ◀

▶ **Example 19.** Consider again the PTA $\mathcal{A}$ in Figure 1a. We have $\mathit{Diff}(\mathcal{A})$ is $(0 \leq p_1 \leq 3 < d \leq p_2) \vee (0 \leq d \leq 3 \wedge d < p_1 \wedge p_2 \geq 0) \vee (0 \leq p_2 < d \leq 3 \wedge p_1 \geq 0)$. Then $\mathit{Diff}(\mathcal{A})\downarrow_{\mathbb{P}}$ is $(0 \leq p_1 \leq 3 < p_2) \vee (0 < p_1 \wedge p_2 \geq 0) \vee (0 \leq p_2 < 3 \wedge p_1 \geq 0)$. Hence, $\texttt{d-FOS}(\mathcal{A})$ is $p_1 = 0 \leq d \leq p_2 = 3$.

Finally, obtaining $\exists\texttt{OS}(\mathcal{A})$ and $\texttt{FOS}(\mathcal{A})$ is trivial since, by definition, $\exists\texttt{OS}(\mathcal{A}) = (\texttt{d-}\exists\texttt{OS}(\mathcal{A}))\downarrow_{\mathbb{P}}$ and $\texttt{FOS}(\mathcal{A}) = (\texttt{d-FOS}(\mathcal{A}))\downarrow_{\mathbb{P}}$.

▶ **Example 20.** Consider again the PTA $\mathcal{A}$ in Figure 1a. Then $\exists\texttt{OS}(\mathcal{A})$ is $0 \leq p_1 \leq p_2 \wedge p_1 \leq 3$. And $\texttt{FOS}(\mathcal{A})$ is $p_1 = 0 \wedge p_2 = 3$.

On correctness and termination

We described here a method for computing $\exists\texttt{OS}(\mathcal{A})$ and $\texttt{FOS}(\mathcal{A})$ for a PTA, that produces an exact (sound and complete) result if it terminates. It relies on the PET of two subsets of the PTA, the computation of which requires enrichment with one clock and one parameter. If they can be computed, those PET take the form of a finite union of convex polyhedra, on which are then applied the union, intersection, difference and projection set operations – that are known to be decidable in this context. Thus the actual termination of the whole semi-algorithm relies on the reachability synthesis of two $(n+1, 0, m+1)$-PTAs. Reachability synthesis is known to be effectively computable for $(1, 0, m)$-PTAs [10], and cannot be achieved for PTAs with 3 parametric clocks or more due to the undecidability of the reachability emptiness problem [2]. For the semi-algorithm we proposed here for $\exists\texttt{OS}$ and $\texttt{FOS}$ problems, we therefore do not have any guarantees of termination, even with only one parametric clock (due to the additional clock x_{abs}), although this might change depending on future results regarding the decidability of reachability synthesis for PTAs with 2 parametric clocks (a first decidability result for the emptiness only was proved for $(2, *, 1)$-PTAs over discrete time [21]).

4 Decidability and undecidability of FOE for 1-clock-PTAs

In this section, we:

1. propose a method to compute potentially infinite PET on $(1, 0, *)$-PTAs, i.e., PTAs with 1 parametric clock and arbitrarily many parameters (Section 4.1);
2. prove decidability of the $\texttt{FOE}$ problem for $(1, 0, 1)$-PTAs, by rewriting infinite PET in a variant of Presburger arithmetic (Section 4.2);
3. prove undecidability of the $\texttt{FOE}$ problem for $(1, 0, *)$-PTAs (Section 4.2).

4.1 Encoding infinite PET for $(1, 0, *)$-PTAs

Given a PTA $\mathcal{A}$ with exactly 1 clock, the goal of the method described here is to guarantee termination of the computation of $PET(\mathcal{A})$ with an exact result. If the partial solution given in Section 3.1 is applied, it amounts to a reachability synthesis on a PTA with 2 clocks, without guarantee of termination. The gist of this method is a form of divide and conquer, where we solve sub-problems, specifically reachability synthesis on sub-parts of $\mathcal{A}$ without adding an additional clock. The first step consists of building some reset-free PTAs, each representing a meaningful subset of the paths joining two given locations in $\mathcal{A}$. $PET(\mathcal{A})$ is then obtained by combining the results of reachability synthesis performed on those reset-free PTAs. The result is encoded in a (finite) regular expression that represents an infinite union of convex polyhedra. Note that this method works perfectly for rational-valued parameters.

4.1.1 Defining the set of reset-free PTAs

Each of the PTAs we build describes parts of the behavior between two locations. More precisely, they represent all the possible paths such that clock resets may occur only on the last transition of the path. We first define the set of locations that we may need based on whether they are initial, final, or reached by a transition associated to a reset.

▶ **Definition 21** (Final-reset paths $FrP(\mathcal{A}, \ell_f)$)**.** *Let $\mathcal{A}$ be a 1-clock PTA, ℓ_0 its initial location and ℓ_f a location of $\mathcal{A}$. We define as $FrP(\mathcal{A}, \ell_f)$ the set of pairs of locations s.t. $\forall(\ell_i, \ell_j) \in FrP(\mathcal{A}, \ell_f)$*

- *$\ell_i = \ell_0$, or $\ell_i \neq \ell_f$ and there is a clock reset on an incoming edge to ℓ_i,*
- *$\ell_j = \ell_f$, or there is a clock reset on an incoming edge to ℓ_j.*

For each pair of states (ℓ_i, ℓ_j) as defined above, we build a reset-free PTA. If the target state ℓ_j is not final (which is a special case), the reset-free PTA models every path going from ℓ_i to ℓ_j and that ends with a reset on its last step. In particular, this ensures that ℓ_j is reached with clock valuation 0.

▶ **Definition 22** (Reset-free PTA $\mathcal{A}(\ell_i, \ell_j)$)**.** *Let $\mathcal{A}$ be a 1-clock PTA, x its unique clock, and ℓ_i, ℓ_j two locations in $\mathcal{A}$. We define as $\mathcal{A}(\ell_i, \ell_j)$ the reset-free PTA obtained from a copy of $\mathcal{A}$ by:*

1. *creating a duplicate ℓ'_j of ℓ_j;*
2. *for all incoming edges (ℓ, g, a, R, ℓ_j) where $R = \emptyset$, removing (ℓ, g, a, R, ℓ_j) and adding an incoming edge (ℓ, g, a, R, ℓ'_j);*
3. *if $\ell_j \neq \ell_\mathrm{f}$, then for all outgoing edges (ℓ_j, g, a, R, ℓ), removing (ℓ_j, g, a, R, ℓ) and adding an outgoing edge (ℓ'_j, g, a, R, ℓ),*
   else, making ℓ'_j urgent and adding an edge $(\ell'_j, \mathit{True}, \epsilon, \emptyset, \ell_j)$;
4. *removing any upper bound invariant on ℓ_j and making it urgent;*
5. *if $\ell_i \neq \ell_j$, setting ℓ_i as the initial location,*
   else, setting ℓ'_j as the initial location;
6. *removing any clock reset on incoming edges to ℓ_j and pruning all other edges featuring a clock reset, and all outgoing edges from ℓ_f;*
7. *adding a parameter d, and a guard $x = d$ to all incoming edges to ℓ_j;*

We will show next how the reachability synthesis of those reset-free PTAs corresponds to fragments of the runs that are considered in $\mathit{PET}(\mathcal{A})$. For simplification, given $\mathcal{A}$ a 1-clock PTA, and ℓ_i, ℓ_j two locations of $\mathcal{A}$, we now note $Z_{\ell_i,\ell_j} = \mathsf{EFsynth}(\mathcal{A}(\ell_i, \ell_j), \{\ell_j\})$.

4.1.2 Reconstruction of PET from the reachability synthesis of the reset-free PTAs

Given $\mathcal{A}$ a 1-clock PTA, and ℓ_f a location of $\mathcal{A}$, for all $(\ell_i, \ell_j) \in \mathit{FrP}(\mathcal{A}, \ell_\mathrm{f})$ we may compute the parametric zone Z_{ℓ_i,ℓ_j} with guarantee of termination, since the reachability synthesis is decidable on 1-clock PTAs. Those parametric zones may be used to build the (potentially infinite) PET of $\mathcal{A}$. To do so, we first define a (non-parametric, untimed) finite automaton where the states are the locations of $\mathcal{A}$, and the arc between the states ℓ_i and ℓ_j is labeled by Z_{ℓ_i,ℓ_j}. We refer to this automaton as the *automaton of the zones* of $\mathcal{A}$.

▶ **Definition 23** (Automaton of the zones)**.** *Let $\mathcal{A}$ be a 1-clock PTA, ℓ_0 its initial location and ℓ_f a location of $\mathcal{A}$. We define as $\hat{\mathcal{A}}$ the finite automaton such that:*

- *The states of $\hat{\mathcal{A}}$ are exactly the locations of $\mathcal{A}$;*
- *ℓ_0 is initial and ℓ_f is final;*
- *$\forall (\ell_i, \ell_j) \in \mathit{FrP}(\mathcal{A}, \ell_\mathrm{f})$, there is a transition from ℓ_i to ℓ_j labeled by Z_{ℓ_i,ℓ_j}.*

We claim that the language $\hat{L}$ of $\hat{\mathcal{A}}$ is a representation of the times (along with parameter constraints) to go from ℓ_0 to ℓ_f in $\mathcal{A}$. As $\hat{\mathcal{A}}$ is a finite automaton, $\hat{L}$ can be represented as a regular expression with three operators: the concatenation (.), the alteration (+), and the Kleene star (*). $\mathit{PET}(\mathcal{A})$ can thus be expressed by redefining those operators with operations on the parametric zones that label edges of $\hat{L}$.

Any parametric zone $Z_{a,b}$ labeling an edge of $\hat{\mathcal{A}}$ is of the form $\bigcup_i \mathbf{C}_i$ with $1 \leq i \leq n$ and $\mathbf{C}_i$ a convex polyhedra. As per Definition 6, $\mathbf{C}_i$ is a conjunction of inequalities, each of the form $\alpha d + \sum_{1 \leq i \leq M} \beta_i p_i + \gamma \bowtie 0$, with $p_i \in \mathbb{P}$, and $\alpha, \beta_i, \gamma \in \mathbb{Z}$. Note that x has been replaced by execution times d, as per Definition 11. In the following, we denote by $\mathbf{C}_i^d$ all

inequalities such that $\alpha \neq 0$ (i.e., inequalities over d and possibly some parameters in $\mathbb{P}$), and by $\mathbf{C}_i^{\mathbb{P}}$ all inequalities such that $\alpha = 0$ (i.e., inequalities strictly over $\mathbb{P}$). This means that $\mathbf{C}_i = \mathbf{C}_i^d \wedge \mathbf{C}_i^{\mathbb{P}}$. For simplification of what follows, we write inequalities in $\mathbf{C}_i^d$ as $d \bowtie c$ where $c = \frac{\sum_{1 \leq i \leq M} \beta_i p_i + \gamma}{-\alpha}$.

Given $Z_{a,b} = \bigcup_i \mathbf{C}_i$ and $Z_{c,d} = \bigcup_j \mathbf{C}_j$, we define the operators $\bar{.}$, $\bar{*}$ and $\bar{+}$.

Operator $\bar{.}$ is the addition of the time durations and intersection of parameter constraints between two parametric zones. Formally, $Z_{a,b} \bar{.} Z_{c,d} = \bigcup_{i*j} \mathbf{C}_{i,j}^d \cap \mathbf{C}_{i,j}^{\mathbb{P}}$ such that $\mathbf{C}_{i,j}^{\mathbb{P}} = \mathbf{C}_i^{\mathbb{P}} \wedge \mathbf{C}_j^{\mathbb{P}}$, and for all $d \bowtie c_i \in \mathbf{C}_i^d$ and $d \bowtie' c_j \in \mathbf{C}_j^d$, if $\bowtie, \bowtie' \in \{<, \leq, =\}$ or $\bowtie, \bowtie' \in \{>, \geq, =\}$, then $d \bowtie'' c_i + c_j \in \mathbf{C}_{i,j}^d$ with $\bowtie''$ being in the same direction as $\bowtie$ and $\bowtie'$ and is

- a strict inequality if either $\bowtie$ or $\bowtie'$ is a strict inequality;
- an equality if both $\bowtie$ and $\bowtie'$ are equalities;
- a non-strict inequality otherwise.

Operator $\bar{*}$ is the recursive application of $\bar{.}$ on a parametric zone. Formally, $Z_{a,b}{}^{\bar{*}} = \bigcup_{K \in \mathbb{N}} \{d = 0\}(\bar{.}Z_{a,b})^K$ where $(\bar{.}Z_{a,b})$ is repeated K times, with K being any value in $\mathbb{N}$. Note that $\{d = 0\}$ corresponds to the case where the loop is never taken, and that it is neutral for the $\bar{.}$ operator: $\{d = 0\}\bar{.}Z_{a,b} = Z_{a,b}$. Also note that, in practice, $a = b$ whenever we use this operator.

Operator $\bar{+}$ is the union of two parametric zones. Formally, $Z_{a,b} \bar{+} Z_{c,d} = Z_{a,b} \cup Z_{c,d}$.

Note that the result of any of those operations is a union of convex polyhedra of the form $\bigcup_i \mathbf{C}_i$, meaning that these operators can be nested. Also, this union is infinite whenever operator $\bar{*}$ is present.

▶ **Proposition 24.** *Let $\mathcal{A}$ be a 1-clock PTA and ℓ_{f} a location of $\mathcal{A}$. Let $\hat{L}$ be the language of the automaton of the zones $\hat{\mathcal{A}}$, and e a regular expression describing $\hat{L}$. Let $\bar{e}$ be the expression obtained by replacing the ., + and * operators in e respectively by $\bar{.}$, $\bar{+}$ and $\bar{*}$. We have $\bar{e} = PET(\mathcal{A})$.*

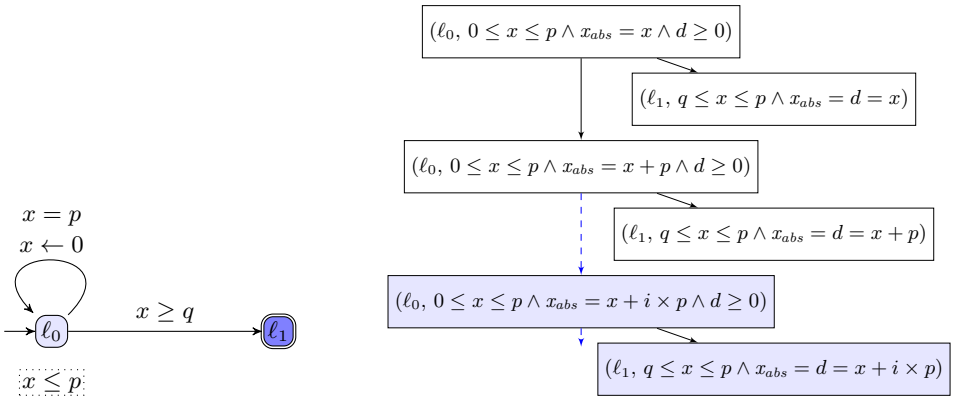
Proof. See Section B.4. ◀

4.1.3 Summary and illustration of the encoding

Given a PTA $\mathcal{A}$ with exactly 1 clock, and given a location ℓ_{f} of $\mathcal{A}$, we compute with an exact result an encoding of $PET(\mathcal{A})$, through the following steps:

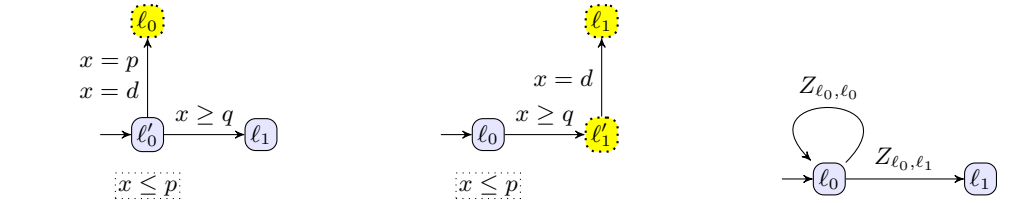
1. compute $FrP(\mathcal{A}, \ell_{\mathrm{f}})$, the pairs of locations (ℓ_i, ℓ_j) such that on some run from initial location to ℓ_{f} there might exists a sub-path from ℓ_i to ℓ_j, such that the clock is reset when entering both locations, but never in between;
2. for each of those pairs, compute the reset-free PTA $\mathcal{A}(\ell_i, \ell_j)$, for which reachability synthesis, noted Z_{ℓ_i,ℓ_j} corresponds to the aforementioned sub-paths;
3. generate the automaton of the zones $\hat{\mathcal{A}}$, on which each pair of locations (ℓ_i, ℓ_j) is connected by a transition labeled with Z_{ℓ_i,ℓ_j};
4. compute a regular expression for $\hat{\mathcal{A}}$, which we proved to be equivalent to $PET(\mathcal{A})$. Note that computing a regular expression from a finite automaton is decidable and there exists numerous efficient methods for this [20].

Before discussing how this regular expression can be used to answer the Full ET-opacity p emptiness problem, let us illustrate how it is obtained on a simple example. Figure 2a depicts a 1-clock PTA $\mathcal{A}$ with a clock x and two parameters p and q. We are interested in solving $PET(\mathcal{A})$ where we assume here that ℓ_{f} is ℓ_1. Applying the semi-algorithm from Section 3.1, suppose the addition of a clock x_{abs} and parameter d to the PTA, followed by the computation of the reachability synthesis to ℓ_1. In this case, the algorithm does not terminate though, and as shown in Figure 2b.



(a) 1-clock PTA $\mathcal{A}$. **(b)** Symbolic (infinite) state space when computing PET in a naive way.

Figure 2 A 1-clock PTA and the PET problem.



(a) $\mathcal{A}(\ell_0, \ell_0)$. **(b)** $\mathcal{A}(\ell_0, \ell_1)$. **(c)** Automaton of the zones $\hat{\mathcal{A}}$.

Figure 3 Reset-free automata of $\mathcal{A}$ (from Figure 2a) and automaton of the zones $\hat{\mathcal{A}}$.

Following the steps of our method, we have $FrP(\mathcal{A}, \ell_1) = \{(\ell_0, \ell_0), (\ell_0, \ell_1)\}$. Figures 3a and 3b depict the corresponding reset-free automata while Figure 3c gives the automaton of the zones. Urgent locations are colored in yellow.

Reachability synthesis of the reset-free automata gives $Z_{\ell_0,\ell_0} = \{d = p\}$ and $Z_{\ell_0,\ell_1} = \{q \leq d \leq p\}$. As per Proposition 24, the expression $\bar{e}$ (obtained by replacing operators in the regular expression of the language of $\hat{\mathcal{A}}$) is equivalent to $PET(\mathcal{A})$ (again taking ℓ_1 as final location). That expression can be easily obtained (for example with a state elimination method) and gives $\bar{e} = (Z_{\ell_0,\ell_0})^{\bar{*}} \bar{\cdot} Z_{\ell_0,\ell_1}$. We may then develop operations on $\bar{e}$ and obtain the following infinite disjunction of parametric zones.

$$\begin{aligned} PET(\mathcal{A}) &= (Z_{\ell_0,\ell_0})^{\bar{*}} \bar{\cdot} Z_{\ell_0,\ell_1} \\ &= \{d = p\}^{\bar{*}\bar{\cdot}} \{q \leq d \leq p\} \\ &= \{d = 0 \vee d = p \vee d = 2p \vee \dots\}.\{q \leq d \leq p\} \\ &= \{q \leq d \leq p \vee q + p \leq d \leq 2p \vee q + 2p \leq d \leq 3p \vee \dots\} \end{aligned} \tag{1}$$

4.2 Solving the FOE problem through a translation of PET to parametric Presburger arithmetic

Presburger arithmetic is the first order theory of the integers with addition. It is a useful tool that can represent and manipulate sets of integers called semi-linear sets. Those sets are particularly meaningful to study TAs, as the set of durations of runs reaching the final location can be described by a semi-linear set [14]. Presburger arithmetic is however not expressive enough to represent durations of runs in PTAs due to the presence of parameters. In [27], a

parametric extension of Presburger arithmetic was considered, introducing linear parametric semi-linear sets (LpSl sets) which are functions associating to a parameter valuation v a (traditional) semi-linear set of the following form:

$$S(v) = \Big\{ x \in \mathbb{N}^m \mid \bigvee_{i \in I} \exists x_0, \dots x_{n_i} \in \mathbb{N}^m, k_1, \dots k_{n_i} \in \mathbb{N}, x = \sum_{j=0}^{n_i} x_j \\ \wedge b_0^i(v) \leq x_0 \leq c_0^i(v) \wedge \bigwedge_{j=1}^{n_i} k_j b_j^i(v) \leq x_j \leq k_j c_j^i(v) \Big\} \quad (2)$$

where I is a finite set and the b_j^i and c_j^i are affine functions with coefficients in $\mathbb{N}$. A 1-LpSl set is an LpSl set defined over a single parameter. Given two LpSl (resp. 1-LpSl) sets S_1 and S_2, the LpSl (resp. 1-LpSl) equality problem consists in deciding whether there exists a parameter valuation v such that $S_1(v) = S_2(v)$.

▶ **Theorem 25** ([27]). *The LpSl equality problem is undecidable.*

The 1-LpSl equality problem is decidable. Moreover, the set of valuations achieving equality can be computed.

The main goal of this subsection is to relate the expressions computed in Section 4.1 to LpSl sets in order to tackle ET-opacity problems. Since Presburger arithmetic is a theory of integers, we have to restrict PTAs to integer parameters; this is what prevents our results to be extended to rational-valued parameters in a straightforward manner. Moreover, we need to focus on time durations of runs with integer values. This second restriction however is without loss of generality. Indeed, in [8, Theorem 5], a trick is provided (which consists mainly in doubling every term of the system so that any run duration that used to be a rational of the form $\frac{q}{2}$ is now an integer to ensure that if a set is non-empty, it contains an integer. This transformation also allows one to consider only non-strict constraints, and thus we assume every constraint is non-strict in the following.

▶ **Theorem 26.** *The LpSl equality problem reduces to the* `FOE` *problem for* $(1, 0, *)$*-PTAs.*

Moreover, the `FOE` *problem for* $(1, 0, 1)$*-PTAs reduces to the 1-LpSl equality problem.*

Sketch of proof. From Equation (2) one can see that an LpSl set parametrically defines integers that are the sum of two types of elements: x_0 belongs to an interval, while the x_j represent a sum of integers, each coming from the interval $[b_j^i; c_j^i]$. Intuitively, we separate a run into its elementary path until the final state and its loops. We use x_0 to represent the duration of the elementary path, and the x_j adds the duration of loops. Each occurrence of the same loop within a run being independent (as they include a reset of the clock), their durations all belong to the same interval.

Formally, given a PTA $\mathcal{A}$, using Section 3.2, we build the PTAs $\mathcal{A}_{\ell_f}^{\ell_{priv}}$ and $\mathcal{A}_{\ell_f}^{\neg\ell_{priv}}$ separating the private and public runs of $\mathcal{A}$. Then with Section 4.1, we obtain expressions $\bar{e}_{\ell_{priv}}$ and $\bar{e}_{\neg\ell_{priv}}$ such that (Proposition 24) $\bar{e}_{\ell_{priv}} = PET(\mathcal{A}_{\ell_f}^{\ell_{priv}})$ and $\bar{e}_{\neg\ell_{priv}} = PET(\mathcal{A}_{\ell_f}^{\neg\ell_{priv}})$. We then develop and simplify these expressions until we can build LpSl sets representing the integers accepted by each expression. We can then show the inter-reduction as the full ET-opacity is directly equivalent to the equality of the two sets. Note that one direction of the reduction is stronger, allowing multiple parameters. This is due to constraints over the parameters which may appear in our expressions, but cannot be transferred to LpSl sets. However, when there is a single parameter, one can easily resolve these constraints beforehand. See Section B.5 for a complete proof. ◀

Combining Theorems 25 and 26 directly gives us:

▶ **Corollary 27.** *FOE is undecidable for* $(1, 0, *)$*-PTAs.*

▶ **Corollary 28.** *FOE is decidable for* $(1, 0, 1)$*-PTAs and FOS can be solved.*

5 Decidability of ∃OE for $(1, 0, *)$-PTAs for integer-valued parameters

We prove here the decidability of ∃OE for $(1, 0, *)$-PTAs with integer parameters over dense time (Section 5.1); we also prove that the same problem is in EXPSPACE for $(1, *, 1)$-PTAs over discrete time (Section 5.2).

5.1 General case

Adding the divisibility predicate (denoted "|") to Presburger arithmetic produces an undecidable theory, whose purely existential fragment is known to be decidable [26]. The FOE problem can be encoded in this logic, but requires a single quantifier alternation, which goes beyond the aforementioned decidability result, leading us to rely on [27]. The ∃OE problem however can be encoded in the purely existential fragment.

▶ **Theorem 29.** *The ∃OE problem is decidable.*

Sketch of proof. As for Theorem 26, we start by building and simplifying expressions representing the private and public durations of the PTA. Instead of translating the expression into LpSl set however, we now use Presburger with divisibility.

Again, a run can be decomposed in the run without loops, and its looping parts. The duration of the former is defined directly by conjunction of inequalities, which can be formulated in a Presburger arithmetic formula. The latter requires the divisibility operator to represent the arbitrary number of loops. Hence, we can build a formula accepting exactly the integers satisfying our expressions. Deciding the ∃OE problem can be achieved by testing the existence of an integer satisfying the formulas produced from both expressions, which can be stated in a purely existential formula. See [11] for a complete proof. ◀

▶ Remark 30 (complexity). Let us quickly discuss the complexity of this algorithm. The expressions produced by Proposition 24 can, in the worst case, be exponential in the size of the PTA. This formula was then simplified within the proof of Theorem 26, in part by developing it, which could lead to an exponential blow-up. Finally, the existential fragment of Presburger arithmetic with divisibility can be solved in NEXPTIME [26]. As a consequence, our algorithm lies in 3NEXPTIME.

5.2 Discrete time case

There are clear ways to improve the complexity of this algorithm. In particular, we finally prove an alternative version of Theorem 29 in a more restricted setting ($\mathbb{T} = \mathbb{N}$), but with a significantly lower complexity upper bound and using completely different proof ingredients [21].

▶ **Theorem 31.** *∃OE is decidable in* EXPSPACE *for* $(1, *, 1)$*-PTAs over discrete time.*

Proof. See [11]. ◀

▶ Remark 32. The fact that we can handle arbitrarily many non-parametric clocks in Theorem 31 does not improve Theorem 29: over discrete time, it is well-known that non-parametric clocks can be eliminated using a technique from [2], and hence come "for free".

Table 1 Execution-time opacity problems for PTAs: contributions and some open cases.

Time	(pc, npc, p)	∃OE emptiness	∃OE synthesis
dense	$(1, 0, *)$	✓ Th. 29	?
dense	$(1, *, *)$	?	?
dense	$(2, 0, 1)$	?	?
dense	$(3, 0, 1)$	× [9, Th.6.1]	×
discrete	$(1, *, 1)$	✓EXPSPACE Th. 31	?

Time	(pc, npc, p)	FOE emptiness	FOE synthesis
dense	$(1, 0, 1)$	✓ Corol. 28	✓ Corol. 28
dense	$(1, 0, [2, M))$	?	?
dense	$(1, 0, M)$	× Corol. 27	×
dense	$([2, 3], 0, 1)$	?	?
dense	$(4, 0, 2)$	× [9, Th. 7.1]	×

6 Conclusion and perspectives

In this paper, we addressed the ET-opacity for 1-clock PTAs with integer-valued parameters over dense time. We proved that 1) FOE is undecidable for a sufficiently large number of parameters, 2) FOE becomes decidable for a single parameter, and 3) ∃OE is decidable, in 3NEXPTIME over dense time and in EXPSPACE over discrete time. These results rely on a novel construction of PET, for which a sound and complete computation method is provided. In the general case, we provided semi-algorithms for the computation of PET, ∃OS and FOS.

The undecidability result reduces from a problem in parametric Presburger arithmetic, itself reducing from Hilbert's tenth problem. The latter is known to be undecidable for various classes of polynomials (with degree 4 and 58 variables for instance). The number of parameters used in the undecidability of the parametric Presburger arithmetic problem is not direct from their proof but we can estimate that at least 200 parameters are needed. Closing the gap through this approach would require important developments in Diophantine analysis. The opacity problem hence remains open for many cases of low number of parameters.

Our PET constructions and all PET-related results work perfectly for rational-valued parameters. It remains however unclear how to extend our (un)decidability results to rational-valued parameters, as our other proof ingredients (notably using the Presburger arithmetics) heavily rely on integer-valued parameters.

It remains also unclear whether synthesis can be achieved using techniques from [21], explaining the "open" cell in the "discrete time" row of Table 1. Also, a number of problems remain open in Table 1, notably the 2-clock case, already notoriously difficult for reachability emptiness [2, 21].

Finally, exploring *weak* ET-opacity [7] (which allows the attacker to deduce that the private location was *not* visited) is also on our agenda.

References

1 Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, April 1994. doi:10.1016/0304-3975(94)90010-8.

2 Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. Parametric real-time reasoning. In S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal, editors, *STOC*, pages 592–601, New York, NY, USA, 1993. ACM. doi:10.1145/167088.167242.

3 Ikhlass Ammar, Yamen El Touati, Moez Yeddes, and John Mullins. Bounded opacity for timed systems. *Journal of Information Security and Applications*, 61:1–13, September 2021. doi:10.1016/j.jisa.2021.102926.

4 Jie An, Qiang Gao, Lingtai Wang, Naijun Zhan, and Ichiro Hasuo. The opacity of timed automata. In André Platzer, Kristin-Yvonne Rozier, Matteo Pradella, and Matteo Rossi, editors, *FM*, volume 14933 of *Lecture Notes in Computer Science*, pages 620–637. Springer, 2024. doi:10.1007/978-3-031-71162-6_32.

5 Étienne André. What's decidable about parametric timed automata? *International Journal on Software Tools for Technology Transfer*, 21(2):203–219, April 2019. doi:10.1007/s10009-017-0467-0.

6 Étienne André, Sarah Dépernet, and Engel Lefaucheux. The bright side of timed opacity. In Kazuhiro Ogata, Meng Sun, and Dominique Méry, editors, *ICFEM*, 2024. To appear.

7 Étienne André, Engel Lefaucheux, Didier Lime, Dylan Marinho, and Jun Sun. Configuring timing parameters to ensure execution-time opacity in timed automata. In Maurice H. ter Beek and Clemens Dubslaff, editors, *TiCSA*, Electronic Proceedings in Theoretical Computer Science. Springer, 2023. Invited paper.

8 Étienne André, Engel Lefaucheux, and Dylan Marinho. Expiring opacity problems in parametric timed automata. In Yamine Ait-Ameur and Ferhat Khendek, editors, *ICECCS*, pages 89–98, 2023. doi:10.1109/ICECCS59891.2023.00020.

9 Étienne André, Didier Lime, Dylan Marinho, and Jun Sun. Guaranteeing timed opacity using parametric timed model checking. *ACM Transactions on Software Engineering and Methodology*, 31(4):1–36, October 2022. doi:10.1145/3502851.

10 Étienne André, Didier Lime, and Nicolas Markey. Language preservation problems in parametric timed automata. *Logical Methods in Computer Science*, 16(1), January 2020. doi:10.23638/LMCS-16(1:5)2020.

11 Étienne André, Johan Arcile, and Engel Lefaucheux. Execution-time opacity problems in one-clock parametric timed automata (extended version). Technical Report abs/2410.01659, arXiv, October 2024. arXiv:2410.01659.

12 Nikola Beneš, Peter Bezděk, Kim Gulstrand Larsen, and Jiří Srba. Language emptiness of continuous-time parametric timed automata. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *ICALP, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 69–81. Springer, July 2015. doi:10.1007/978-3-662-47666-6_6.

13 Arnab Kumar Biswas, Dipak Ghosal, and Shishir Nagaraja. A survey of timing channels and countermeasures. *ACM Computing Surveys*, 50(1):6:1–6:39, 2017. doi:10.1145/3023872.

14 Véronique Bruyère, Emmanuel Dall'Olio, and Jean-Francois Raskin. Durations and parametric model-checking in timed automata. *ACM Transactions on Computational Logic*, 9(2):12:1–12:23, 2008. doi:10.1145/1342991.1342996.

15 Jeremy W. Bryans, Maciej Koutny, Laurent Mazaré, and Peter Y. A. Ryan. Opacity generalised to transition systems. *International Journal of Information Security*, 7(6):421–435, 2008. doi:10.1007/s10207-008-0058-x.

16 Daniel Bundala and Joël Ouaknine. On parametric timed automata and one-counter machines. *Information and Computation*, 253:272–303, 2017. doi:10.1016/j.ic.2016.07.011.

17 Franck Cassez. The dark side of timed opacity. In Jong Hyuk Park, Hsiao-Hwa Chen, Mohammed Atiquzzaman, Changhoon Lee, Tai-Hoon Kim, and Sang-Soo Yeo, editors, *ISA*, volume 5576 of *Lecture Notes in Computer Science*, pages 21–30. Springer, 2009. doi:10.1007/978-3-642-02617-1_3.

18 Catalin Dima. Real-time automata. *Journal of Automata, Languages and Combinatorics*, 6(1):3–23, 2001. doi:10.25596/jalc-2001-003.

19 Guillaume Gardey, John Mullins, and Olivier H. Roux. Non-interference control synthesis for security timed automata. *Electronic Notes in Theoretical Computer Science*, 180(1):35–53, 2007. doi:10.1016/j.entcs.2005.05.046.

20 Hermann Gruber and Markus Holzer. From finite automata to regular expressions and back - A summary on descriptional complexity. *International Journal of Foundations of Computer Science*, 26(8):1009–1040, 2015. doi:10.1142/S0129054115400110.

21 Stefan Göller and Mathieu Hilaire. Reachability in two-parametric timed automata with one parameter is EXPSPACE-complete. In Markus Bläser and Benjamin Monmege, editors, *STACS*, volume 187 of *LIPIcs*, pages 36:1–36:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.STACS.2021.36.

22 Thomas A. Henzinger, Zohar Manna, and Amir Pnueli. Timed transition systems. In J. W. de Bakker, Cornelis Huizing, Willem P. de Roever, and Grzegorz Rozenberg, editors, *REX*, volume 600 of *Lecture Notes in Computer Science*, pages 226–251. Springer, 1992. doi:10.1007/BFb0031995.

23 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. doi:10.1016/S1567-8326(02)00037-1.

24 Aleksandra Jovanović, Didier Lime, and Olivier H. Roux. Integer parameter synthesis for real-time systems. *IEEE Transactions on Software Engineering*, 41(5):445–461, 2015. doi:10.1109/TSE.2014.2357445.

25 Julian Klein, Paul Kogel, and Sabine Glesner. Verifying opacity of discrete-timed automata. In Nico Plat, Stefania Gnesi, Carlo A. Furia, and Antónia Lopes, editors, *FormaliSE*, pages 55–65. ACM, 2024. doi:10.1145/3644033.3644376.

26 Antonia Lechner, Joël Ouaknine, and James Worrell. On the complexity of linear arithmetic with divisibility. In *LICS*, pages 667–676. IEEE Computer Society, 2015. doi:10.1109/LICS.2015.67.

27 Engel Lefaucheux. When are two parametric semi-linear sets equal? Technical Report hal-04172593, HAL, 2024. URL: https://inria.hal.science/hal-04172593.

28 Laurent Mazaré. Using unification for opacity properties. In Peter Ryan, editor, *WITS*, pages 165–176, April 2004.

29 Joseph S. Miller. Decidability and complexity results for timed automata and semi-linear hybrid automata. In Nancy A. Lynch and Bruce H. Krogh, editors, *HSCC*, volume 1790 of *Lecture Notes in Computer Science*, pages 296–309. Springer, 2000. doi:10.1007/3-540-46430-1_26.

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

31 Lingtai Wang and Naijun Zhan. Decidability of the initial-state opacity of real-time automata. In Cliff B. Jones, Ji Wang, and Naijun Zhan, editors, *Symposium on Real-Time and Hybrid Systems - Essays Dedicated to Professor Chaochen Zhou on the Occasion of His 80th Birthday*, volume 11180 of *Lecture Notes in Computer Science*, pages 44–60. Springer, 2018. doi:10.1007/978-3-030-01461-2_3.

32 Lingtai Wang, Naijun Zhan, and Jie An. The opacity of real-time automata. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(11):2845–2856, 2018. doi:10.1109/TCAD.2018.2857363.

A Recalling the correctness of EFsynth

▶ **Lemma 33** ([24]). *Let $\mathcal{A}$ be a PTA, and let L_{target} be a subset of the locations of $\mathcal{A}$. Assume $\mathsf{EFsynth}(\mathcal{A}, L_{target})$ terminates with result K. Then $v \models K$ iff L_{target} is reachable in $v(\mathcal{A})$.*

B Proof of results

B.1 Proof of Proposition 18

▶ **Proposition 18.** *Given a PTA $\mathcal{A}$ with parameter set $\mathbb{P}$: $\mathtt{d\text{-}FOS}(\mathcal{A}) = \mathtt{d\text{-}\exists OS}(\mathcal{A}) \setminus \mathit{Diff}(\mathcal{A})\downarrow_{\mathbb{P}}$.*

Proof. By definition, $\mathtt{d\text{-}FOS}(\mathcal{A})$ is the synthesis of parameter valuations v (and execution times of their runs) s.t. $v(\mathcal{A})$ is fully opaque w.r.t. ℓ_{priv} on the way to ℓ_{f}. By definition, $\mathit{Diff}(\mathcal{A})\downarrow_{\mathbb{P}}$ is the set of parameter valuations s.t. for any valuation $v \in \mathit{Diff}(\mathcal{A})\downarrow_{\mathbb{P}}$, there is at least one run where ℓ_{priv} is reached (resp. avoided) on the way to ℓ_{f} in $v(\mathcal{A})$ whose duration time is different from those of any run where ℓ_{priv} is avoided (resp. reached) on the way to ℓ_{f} in $v(\mathcal{A})$. By removing this set of parameters from $\mathtt{d\text{-}\exists OS}(\mathcal{A})$, we are left with parameter valuations (and execution times of their runs) s.t. for any v, any run ρ where ℓ_{priv} is reached (resp. avoided) on the way to ℓ_{f} in $v(\mathcal{A})$, there is a run ρ' where ℓ_{priv} is avoided (resp. reached) on the way to ℓ_{f} in $v(\mathcal{A})$ and $dur(\rho) = dur(\rho')$. This is equivalent to our definition of full opacity. ◀

B.2 Proposition 34

▶ **Proposition 34.** *Let $\mathcal{A}$ be a 1-clock PTA, and $(\ell_i, \ell_j) \in FrP(\mathcal{A}, \ell_f)$ such that $\ell_j \neq \ell_f$. Then Z_{ℓ_i,ℓ_j} is equivalent to the synthesis of parameter valuations v and execution times D_v such that $D_v = \{d \mid \exists \rho$ from $(\ell_i, \{x = 0\})$ to ℓ_j in $v(\mathcal{A})$ such that $d = dur(\rho)$, ℓ_f is never reached, and x is reset on the last edge of ρ and on this edge only $\}$.*

Proof. Let us first consider the case where $\ell_i \neq \ell_j$. Steps 1 to 3 in Definition 22 imply that whenever ℓ_j occurs either as a source or target location in an edge, it is replaced by the duplicate locality ℓ'_j, except when ℓ_j is the target location and x is reset on the edge. At this stage, for any path between ℓ_i and ℓ_j in $\mathcal{A}$, where no incoming edge to ℓ_j featuring a clock reset is present, there is an equivalent path in $\mathcal{A}(\ell_i, \ell_j)$ with ℓ_j being replaced by ℓ'_j. Step 4 implies that whenever ℓ_j is reached in $\mathcal{A}(\ell_i, \ell_j)$ no delay is allowed. As there are no outgoings edges from ℓ_j anymore, and only incoming edges featuring a clock reset, only runs ending with such edges are accepted by the reachability synthesis on ℓ_j. Since the clock value when entering in ℓ_j through such an edge is always 0, removing the upper bound of the invariant does not impact the availability of transitions. Because of our assumption that $\ell_i \neq \ell_j$, Step 5 does not change the initial location. Step 6 ensures that, in any run from ℓ_i to ℓ_j :

- no clock reset is performed before the last edge of the run;
- the clock is not reset when entering ℓ_j, and is therefore equals to the duration of the run;
- ℓ_f is not reached.

Step 7 ensures that d is equal to the value of the clock when entering ℓ_f.

Let us now consider the case where $\ell_i = \ell_j$. In this case, Step 5 changes the initial locality to ℓ'_j. Because of Steps 1 to 3, runs from ℓ'_j to ℓ_j in $\mathcal{A}(\ell_i, \ell_j)$ are identical to runs looping from ℓ_i to ℓ_i in $\mathcal{A}$ where x is reset on the last edge of the run and on this edge only. Restrictions obtained by Steps 4, 6 and 7 are unchanged.

Therefore, Z_{ℓ_i,ℓ_j} is equivalent to the synthesis of parameter valuations v and execution times D_v such that $D_v = \{d \mid \exists \rho$ from $(\ell_i, \{x = 0\})$ to ℓ_j in $v(\mathcal{A})$ such that $d = dur(\rho)$, ℓ_f is never reached, and x is reset on the last edge of ρ and on this edge only. ◀

B.3 Proposition 35

▶ **Proposition 35.** *Let $\mathcal{A}$ be a 1-clock PTA, and $(\ell_i, \ell_j) \in FrP(\mathcal{A}, \ell_f)$ such that $\ell_j = \ell_f$. Then Z_{ℓ_i,ℓ_j} is equivalent to the synthesis of parameter valuations v and execution times D_v such that $D_v = \{d \mid \exists \rho$ from $(\ell_i, \{x = 0\})$ to ℓ_f in $v(\mathcal{A})$ such that $d = dur(\rho)$, ℓ_f is reached only on the last state of ρ, and x may only be reset on the last edge of ρ $\}$.*

Proof. By Definition 21, we know that $\ell_i \neq \ell_f$.

Steps 1 to 3 in Definition 22 imply that:

- whenever ℓ_f is the target location of an edge, it is replaced by the duplicate locality ℓ'_j, except when x is reset on the edge;
- once ℓ'_j is reached, no delay is allowed and the only available transition consists in reaching ℓ_f through an empty action ϵ.

At this stage, the only difference between path from ℓ_i to ℓ_f in $\mathcal{A}(\ell_i, \ell_j)$ and $\mathcal{A}$ is that incoming edges to ℓ_f where x is not reset now leads to ℓ'_j, and then to ℓ_f without any added elapsed time. Step 4 implies that whenever ℓ_f is reached in $\mathcal{A}(\ell_i, \ell_j)$ no delay is allowed. As ℓ_f is either entered by the immediate transition from ℓ'_j or feature a clock reset, removing the upper bound of the invariant does not impact the availability of transitions. As $\ell_i \neq \ell_f$, Step 5 does not change the initial location. Step 6 ensures that, in any run from ℓ_i to ℓ_j :

- no clock reset is performed before the last edge of the run (not counting the ϵ edge from ℓ'_j to ℓ_f);
- the clock value is not reset when entering ℓ_f, and is therefore equal to the duration of the run;
- no action can be taken after reaching ℓ_f.

Step 7 ensures that d is equal to the value of the clock when entering ℓ_f.

Therefore, Z_{ℓ_i,ℓ_j} is equivalent to the synthesis of parameter valuations v and execution times D_v such that $D_v = \{d \mid \exists \rho$ from $(\ell_i, \{x = 0\})$ to ℓ_f in $v(\mathcal{A})$ such that $d = dur(\rho)$, ℓ_f is reached only on the last state of ρ, and x may only be reset on the last edge of ρ. ◀

B.4 Proof of Proposition 24

▶ **Proposition 24.** *Let $\mathcal{A}$ be a 1-clock PTA and ℓ_f a location of $\mathcal{A}$. Let $\hat{L}$ be the language of the automaton of the zones $\hat{\mathcal{A}}$, and e a regular expression describing $\hat{L}$. Let $\bar{e}$ be the expression obtained by replacing the $.$, $+$ and * operators in e respectively by $\bar{.}$, $\bar{+}$ and $\bar{^*}$. We have $\bar{e} = PET(\mathcal{A})$.*

Proof. Let us first show that $\bar{e}$ contains $PET(\mathcal{A})$. Let ρ be a path whose time duration and parameter constraints are in $PET(\mathcal{A})$. By definition, ρ starts at time 0 in the initial locality and ends in ℓ_f, with only one occurrence of ℓ_f in the whole path. Let us consider that the clock is reset n times before the last transition, then ρ can be decomposed as $\rho_0 \dots \rho_n$ such that:

- $\forall\ 0 \leq i < n$, sub-path ρ_i starts in ℓ_i at time valuation 0, ends in ℓ_{i+1}, contains a single reset positioned on the last transition (thus ending with time valuation 0) and does not contain any occurrence of ℓ_f;
- sub-path ρ_n starts in ℓ_n at time valuation 0, ends in ℓ_f, may only contain a reset on its last transition, and contains exactly one occurrence of ℓ_f.

By Definition 21, $\forall\ 0 \leq i < n$, $(\ell_i, \ell_{i+1}) \in FrP(\mathcal{A}, \ell_f)$ and by Proposition 34, $Z_{\ell_i,\ell_{i+1}}$ is the synthesis of parameter valuations and execution times of that sub-path. By Definition 21, $(\ell_n, \ell_f) \in FrP(\mathcal{A}, \ell_f)$ and by Proposition 35, Z_{ℓ_n,ℓ_f} is the synthesis of parameter and valuation times of that sub-path. By Definition 23, there is a sequence of transitions $Z_{\ell_0,\ell_1}, \dots, Z_{\ell_i,\ell_{i+1}}, \dots, Z_{\ell_n,\ell_f}$ in the automaton of the zones $\hat{\mathcal{A}}$. By application of operators $\bar{+}$ and $\bar{^*}$, that sequence thus exists in $\bar{e}$ as $Z_{\ell_0,\ell_1} \bar{.} \dots \bar{.} Z_{\ell_i,\ell_{i+1}} \bar{.} \dots \bar{.} Z_{\ell_n,\ell_f}$. By definition of operator $\bar{.}$, this expression is the intersection of all parameter constraints and the addition of all valuation times, which is equivalent to $PET(\mathcal{A})$.

Let us now show that $PET(\mathcal{A})$ contains $\bar{e}$. By application of operators $\bar{+}$ and $\bar{^*}$, any word in $\bar{e}$ can be expressed as a sequence of concatenation operations $\bar{.}$. By Definition 23, given a word $Z_{\ell_0,\ell_1} \bar{.} \dots \bar{.} Z_{\ell_i,\ell_{i+1}} \bar{.} \dots \bar{.} Z_{\ell_n,\ell n+1} \in \bar{e}$, we know that ℓ_0 is the initial location of $\mathcal{A}$, $\ell_{n+1} = \ell_f$ and $\forall\ 0 \leq i \leq n, \ell_i \neq \ell_f$. By Proposition 34, $\forall\ 0 \leq i < n$, $Z_{\ell_i,\ell_{i+1}}$ is the synthesis of parameter valuations and execution times of paths between ℓ_i and ℓ_{i+1} in $\mathcal{A}$ such that ℓ_f is never reached, and x is reset on the last edge of the path and on this edge only. And by Proposition 35, Z_{ℓ_n,ℓ_f} is the synthesis of parameter valuations and execution times of paths between ℓ_n and ℓ_f in $\mathcal{A}$ such that ℓ_f is reached only on the last state of ρ, and x may only be reset on the last edge of ρ.

Let us assume there exists a path ρ whose time duration and parameter constraints are in $PET(\mathcal{A})$ such that $\rho = \rho_0 \dots \rho_n$ and:

- $\forall\ 0 \leq i < n$, sub-path ρ_i starts in ℓ_i at time valuation 0, ends in ℓ_{i+1}, contains a single reset positioned on the last transition (thus ending with time valuation 0) and does not contain any occurrence of ℓ_f;
- sub-path ρ_n starts in ℓ_n at time valuation 0, ends in ℓ_f, may only contain a reset on its last transition, and contains exactly one occurrence of ℓ_f.

Then $Z_{\ell_0,\ell_1} \bar{.} \dots \bar{.} Z_{\ell_i,\ell_{i+1}} \bar{.} \dots \bar{.} Z_{\ell_n,\ell_{n+1}} \in PET(\mathcal{A})$. On the other hand, if there does not exist such a path, then there exist $0 \leq i \leq n$ such that $Z_{\ell_i,\ell_{i+1}} = \emptyset$. By recursive applications of operator $\bar{.}$, the whole sequence is evaluated as $\emptyset$ and thus contained in $PET(\mathcal{A})$. ◀

B.5 Proof of Theorem 26

▶ **Theorem 26.** *The LpSl equality problem reduces to the* FOE *problem for* $(1, 0, *)$*-PTAs. Moreover, the* FOE *problem for* $(1, 0, 1)$*-PTAs reduces to the 1-LpSl equality problem.*

Proof. Given a PTA $\mathcal{A}$, we showed in Section 3.2 how to compute two PTAs $\mathcal{A}^{\ell_{priv}}_{\ell_f}$ and $\mathcal{A}^{\neg\ell_{priv}}_{\ell_f}$ separating the private and public runs of $\mathcal{A}$. Then in Section 4.1, we showed how to build expressions $\bar{e}_{\ell_{priv}}$ and $\bar{e}_{\neg\ell_{priv}}$ such that (Proposition 24) $\bar{e}_{\ell_{priv}} = PET(\mathcal{A}^{\ell_{priv}}_{\ell_f})$ and $\bar{e}_{\neg\ell_{priv}} = PET(\mathcal{A}^{\neg\ell_{priv}}_{\ell_f})$.

Note that the operators $\bar{.}$, $\bar{*}$ and $\bar{+}$ are associative and commutative; moreover, each term Z occurring in the expressions $\bar{e}_{\ell_{priv}}$ and $\bar{e}_{\neg\ell_{priv}}$ is a union of constraints $Z = \bigcup_{i'} \mathbf{C}_{i'} = \overline{\bigdash}_{i'} \mathbf{C}_{i'}$. As a consequence, we can thus develop the entire expression to the form

$$\overline{\bigdash}_i \, (\mathbf{C}^i_1 \bar{.} \mathbf{C}^i_2 \bar{.} \dots \bar{.} \mathbf{C}^i_{n_i}) \bar{.} (\mathbf{C}^i_{n_i+1})^{\bar{*}} \bar{.} (\mathbf{C}^i_{n_i+2})^{\bar{*}} \bar{.} \dots \bar{.} (\mathbf{C}^i_{n_i+m_i})^{\bar{*}}.$$

where we put all $\bar{+}$ outside of the expression. For example, the expression $Z_1 \bar{.} (Z_2)^{\bar{*}}$ where $Z_1 = \mathbf{C}_1 \cup \mathbf{C}_2$ and $Z_2 = \mathbf{C}_3 \cup \mathbf{C}_4$ is developed into $\mathbf{C}_1 \bar{.} (\mathbf{C}_3)^{\bar{*}} \bar{.} (\mathbf{C}_4)^{\bar{*}} \bar{+} \mathbf{C}_2 \bar{.} (\mathbf{C}_3)^{\bar{*}} \bar{.} (\mathbf{C}_4)^{\bar{*}}$.

As $\mathbf{C}^{\bar{*}} = \{d = 0\} \bar{+} \mathbf{C} \bar{.} \mathbf{C}^{\bar{*}}$, for each $\mathbf{C}^i_{n_i+j}$ we can w.l.o.g. express term i as the union of two terms: one where $(\mathbf{C}^i_{n_i+j})^{\bar{*}}$ is removed (i.e., this loop is never taken), and one where $\mathbf{C}^i_{n_i+j}$ is concatenated to the term (i.e., the loop is taken at least once). This means that each term, is turned into 2^{m_i} terms, where we can assume w.l.o.g. that for each $j > 0$, $\mathbf{C}^i_{n_i+j} = \mathbf{C}^i_j$.

Given an expression of the above form, by definition of $\bar{.}$, the product $\mathbf{C}^i_1 \bar{.} \mathbf{C}^i_2 \bar{.} \dots \bar{.} \mathbf{C}^i_{n_i}$ is also a conjunction of inequalities and thus can be expressed as $\mathbf{C}^d_i \wedge \mathbf{C}^{\mathbb{P}}_i$ where $\mathbf{C}^{\mathbb{P}}_i$ is obtained by the constraints that do not involve d while $\mathbf{C}^d_i$ contains the constraints that involve d and potentially some parameters in $\mathbb{P}$. Note also that by the assumption that for each $j > 0$, $\mathbf{C}^i_{n_i+j} = \mathbf{C}^i_j$, any constraint that does not involve d can be removed from $\mathbf{C}^i_{n_i+j}$ without modifying the set. Therefore, the expression can now be rewritten as

$$\overline{\bigdash}_i (\mathbf{C}^d_i \wedge \mathbf{C}^{\mathbb{P}}_i) \bar{.} (\mathbf{C}^i_1)^{\bar{*}} \bar{.} (\mathbf{C}^i_2)^{\bar{*}} \bar{.} \dots \bar{.} (\mathbf{C}^i_{m_i})^{\bar{*}}.$$

where every inequality in $\mathbf{C}^i_j$ involves d.

- Assume the expressions involve a single parameter p. Let us show that the FOE problem for PTAs over a single parameter reduces to the 1-LpSl equality problem.
  Every constraint on p is of the form $p \bowtie c$ with $c \in \mathbb{N}$ and $\bowtie \in \{\leq, \geq\}$. Therefore, there exists a constant M such that for all i, either the constraint $\mathbf{C}^{\mathbb{P}}_i$ is satisfied for all $p \geq M$, or it is satisfied by none.
  For any fixed valuation v, full ET-opacity of $v(\mathcal{A})$ is decidable by [7]. We thus assume that we consider only valuations of p greater than M. This can be represented by replacing every occurrence of p in the expressions by $M + p$. This can be done without loss of generality as we can independently test whether the PTA is fully ET-opaque for the finitely many integer values of p smaller than M. When solving the FOS problem, we thus need to include the valuations of p smaller than M that achieved equality to the valuations provided by the reduction.

The terms $\mathbf{C}_i^{\mathbb{P}}$ being either always or never valid, one can either remove this constraint from the expression, or the term containing it producing an expression of the form

$$\overline{\bigdash}_i \mathbf{C}_0^i \bar{\cdot} (\mathbf{C}_1^i)^{\bar{*}} \bar{\cdot} (\mathbf{C}_2^i)^{\bar{*}} \bar{\cdot} \cdots \bar{\cdot} (\mathbf{C}_{m_i}^i)^{\bar{*}}.$$

where every constraint involves x.

Once again, assuming p is large enough, the constraint $\mathbf{C}_j^i$ can be assumed to be of the form $\alpha_j^i p + \beta_j^i \leq x \leq \gamma_j^i p + \delta_j^i$ where $\alpha_j^i, \beta_j^i, \gamma_j^i, \delta_j^i \in \mathbb{N}$.

For both expressions $\bar{e}_{\ell_{priv}}$ and $\bar{e}_{\neg \ell_{priv}}$, now in the simplified form described above, we build the 1-LpSl sets $S_{\bar{e}_{\ell_{priv}}}$ and $S_{\bar{e}_{\neg \ell_{priv}}}$ where, taking the notations from Equation (2), I is the set $\overline{\bigdash}$ ranges over, for $0 \leq j \leq m_i, b_j^i = \alpha_j^i p + \beta_j^i$ and $c_j^i = \gamma_j^i p + \delta_j^i$.

For a valuation v of p, we have that $S_{\bar{e}_{\ell_{priv}}}(v)$ contains exactly the integers that satisfy $v(\bar{e}_{\ell_{priv}})$ (and similarly for $S_{\bar{e}_{\neg \ell_{priv}}}(v)$ and $v(\bar{e}_{\neg \ell_{priv}})$). Therefore, there exists a valuation such that $\mathcal{A}$ if fully opaque w.r.t. ℓ_{priv} on the way to ℓ_{f} iff there exists a parameter valuation v such that $S_{\bar{e}_{\ell_{priv}}}(v) = S_{\bar{e}_{\neg \ell_{priv}}}(v)$, establishing the reduction.

- We now wish to show that the LpSl equality problem reduces to the `FOE` problem.

To do so, we fix two LpSl sets S_1 and S_2, then build two automata $\mathcal{A}_1$ and $\mathcal{A}_2$ such that $S_i(v)$ contains exactly the integers that satisfy $v(PET(\mathcal{A}_i))$, for all valuation v, for $i \in \{1, 2\}$.

Let us focus on S_1 and assume it is of the form given by Equation (2). We build $\mathcal{A}_1$ so that from the initial location ℓ_0 it can take multiple transitions (one for each $i \in I$), the ith transition being allowed if the clock lies between b_0^i and c_0^i, reset the clock and reach a state ℓ_i. From ℓ_i, there are n_i loops, and the jth loop can be taken if the clock lies between b_j^i and c_j^i and resets the clock. Moreover, a transition can be taken from ℓ_i to ℓ_{f} if $x = 0$.

Formally, $\mathcal{A}_1 = (\Sigma, L, \ell_0, \mathbb{X}, \mathbb{P}, I, E)$ where $\Sigma = \{\epsilon\}$, $L = \{\ell_0, \ell_{\mathrm{f}}\} \cup \{\ell_i \mid i \in I\}$, $\mathbb{X} = \{x\}$, $\mathbb{P}$ is the set of parameters appearing in S_1, I does not restrict the PTA (i.e., it associates $\mathbb{R}_{\geq 0}$ to every location), and finally

$$\begin{aligned} E = &\big\{(\ell_0, (b_0^i \leq x \leq c_0^i), \epsilon, \{x\}, \ell_i \mid i \in I\big\} \\ &\cup \big\{(\ell_i, (b_j^i \leq x \leq c_j^i), \epsilon, \{x\}, \ell_i \mid i \in I, 1 \leq j \leq n_i\big\} \\ &\cup \big\{(\ell_i, (x = 0), \epsilon, \emptyset, \ell_{\mathrm{f}} \mid i \in I\big\}. \end{aligned}$$

Thus, a run reaching ℓ_{f} can be decomposed into final-reset paths. In other words, there is a run reaching ℓ_{f} with duration d iff d can be written as a sum $d = \sum_{j=0}^{n_i} d_j$ where $b_0^i \leq d_0 \leq c_0^i$ and for all $j > 0$, $k_j b_j^i \leq d_j \leq k_j c_j^i$ where k_j is the number of times the jth loop is taken in the PTA. As a consequence, the set of durations of runs reaching ℓ_{f} is exactly S_1.

We build $\mathcal{A}_2$ similarly. We now build the PTA $\mathcal{A}$ which can either immediately (with $x = 0$) go to the initial state of $\mathcal{A}_1$ or go immediately to a private location ℓ_{priv} before immediately reaching the initial state of $\mathcal{A}_2$. The final location of $\mathcal{A}_1$ and $\mathcal{A}_2$ are then fused in a single location ℓ_{f}. We thus have that, the set of runs reaching ℓ_{priv} on the way to ℓ_{f} are exactly the ones reaching ℓ_{f} in $\mathcal{A}_2$ (with a prefix of duration 0). And similarly, the set of runs avoiding ℓ_{priv} on the way to ℓ_{f} are exactly the ones reaching ℓ_{f} in $\mathcal{A}_1$ (with a prefix of duration 0). Therefore, for any parameter valuation v, we have that $DVisit^{priv}(v(\mathcal{A})) = DVisit^{\overline{priv}}(v(\mathcal{A}))$ iff $S_1(v) = S_2(v)$, concluding the reduction. ◀