Opacity problems in subclasses of timed automata

Abstract. In 2009, Cassez showed that the timed opacity problem, where an attacker can observe some actions with their timestamps and attempts to deduce information, is undecidable for timed automata. Moreover, he showed that the undecidability holds even for subclasses such as event-recording automata. In this article, we consider the same definition of opacity for several other subclasses of timed automata: with restrictions on the number of clocks, of actions, on the nature of time, on a new subclass called observable event-recording automata, or on the number of observations made by the attacker. We show that opacity can mostly be retrieved, except for the notable subclass of one-action timed automata, for which undecidability remains.

1 Introduction

The notion of *opacity* [17,13] 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 [16] that it is possible for models that are opaque when timing constraints are omitted, to become non-opaque when those constraints are added to the models.

Timed automata (TAs) [1] are an extension of finite automata that can measure and react to the passage of time, extending traditional finite automata with the ability to handle real-time constraints. They are equipped with a finite set of clocks that can be reset and compared with integer constants, enabling the modeling and verification of real-time systems.

Related works There are several ways to define opacity problems in TAs, depending on the power of the attacker. The common idea is to ensure that the attacker cannot deduce from the observation of a run whether it was a private or a public run. The attacker in [14] is able to observe a subset $\Sigma_0 \subseteq \Sigma$ of actions with their timestamps. In this context, a timed word w is said to be opaque if there exists a public run that produces the projection of w following Σ_0 as an observed timed word. In this configuration, one can consider the opacity problem consisting of determining, knowing a TA $\mathcal{A}$ and a set of timed words, whether all words in this set are opaque in $\mathcal{A}$. This problem has been shown to be undecidable for TAs [14]. This notably relates to the undecidability of timed language inclusion for TAs [1]. However, the undecidability holds in [14] even for the restricted class of event-recording automata (ERAs) [2] (a subclass of TAs), for which language inclusion is decidable. The aforementioned negative results leave hope only if the definition or the setting is changed, which was done in three main lines of works.

First, in [19,20], the input model is simplified to *real-time automata* [15], a restricted formalism compared to TAs. In this setting, (initial-state) opacity becomes decidable [19,20].

Second, in [4], the authors consider a time-bounded notion of the opacity of [14], where the attacker has to disclose the secret before an upper bound, using a partial observability. This can be seen as a secrecy with an *expiration date*. In addition, the analysis is carried over a time-bounded horizon. The authors prove that this problem is decidable for TAs.

Third, in [8,7], the authors present an alternative definition to Cassez' opacity by studying *execution-time opacity*: the attacker has only access to the execution time of the system, as opposed to Cassez' partial observations with some observable events (with their timestamps). In that case, most problems become decidable (see [6] for a survey).

Orthogonal directions of research include non-interference for TAs, with some decidability results [10,11,5], while control was considered in [12]. General security problems for TAs are surveyed in [9].

Contributions Considering the negative results from [14] we have mainly two directions: one can consider more restrictive classes of automata, or one can limit the capabilities of the attacker—we address both directions in this work.

We address here ∃-opacity ("there exists a pair of runs visiting and not visiting the private locations set, that cannot be distinguished"), weak opacity ("for any run visiting the private locations set, there is another run not visiting it and both cannot be distinguished") and full opacity (weak opacity, with the other direction holding as well). Throughout the first part of this paper (Section 5), we choose to consider the same attacker settings as in [14] but for subclasses of TAs: first we deal with one-clock TAs, then one-action TAs, TAs over discrete time, and a new subclass which we call observable ERAs. Then, in the second part (Section 6), we change our approach and reduce the visibility of the attacker to a *finite* number of actions occurring at the beginning of the run, on an unrestricted TA. In both settings, the attacker knows the TA modeling the system and can observe (some) actions, but does never gain access to the values of the clocks, nor knows which locations are visited. Their goal is to deduce from these observations whether a private location was visited.

We show that:

1. The problem of ∃-opacity is decidable for general TAs and thus for the subclasses of TAs we consider as well (Section 5.1).
2. The problems of weak and full opacity are both undecidable for TAs with only one action or two clocks (Section 5.2) but are decidable for TAs with only one clock (Section 5.3), for unrestricted TAs over discrete time (Section 5.4), and for observable ERAs (Section 5.5).
3. The problems of weak and full opacity are decidable whenever the attacker is restricted to only a finite number of observations (Section 6).

As proof ingredients, we also show that 1) language inclusion is decidable for TAs over discrete time (an unsurprising result, of which we could not find a proof in the literature) and 2) weak opacity and full opacity are inter-reducible.

Outline Section 2 recalls necessary preliminaries. Section 3 defines the problems of interest. Section 4 introduces common constructions used in Sections 5 and 6. Section 5 addresses opacity for subclasses of TAs, while Section 6 reduces the power of the attacker to a finite set of observations. Section 7 concludes.

2 Preliminaries

We denote by $\mathbb{N}, \mathbb{Z}, \mathbb{Q}_{\geq 0}, \mathbb{R}_{\geq 0}$ the sets of non-negative integers, integers, non-negative rationals and non-negative reals, respectively.

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. Throughout this paper, we assume a set $\mathbb{X} = \{x_1, \dots, x_H\}$ of *clocks*. A *clock valuation* is a function $\mu : \mathbb{X} \to \mathbb{T}$, assigning a non-negative value to each clock. We write $\mathbf{0}$ for the clock valuation assigning 0 to all clocks. Given a constant $d \in \mathbb{T}$, $\mu + d$ denotes the valuation s.t. $(\mu + d)(x) = \mu(x) + d$, for all $x \in \mathbb{X}$. If R is a subset of $\mathbb{X}$ and μ a clock valuation, we call *reset* of the clocks of R and denote by $[\mu]_R$ the valuation s.t. for all clock $x \in \mathbb{X}$, $[\mu]_R(x) = 0$ if $x \in R$ and $[\mu]_R(x) = \mu(x)$ otherwise.

We assume $\bowtie \in \{<, \leq, =, \geq, >\}$. A constraint C is a conjunction of inequalities over $\mathbb{X}$ of the form $x \bowtie d$, with $d \in \mathbb{Z}$. Given C, we write $\mu \models C$ if the expression obtained by replacing each x with $\mu(x)$ in C evaluates to true.

Timed automata A TA is a finite automaton extended with a finite set of real-valued clocks. We also add to the standard definition of TAs a special private locations set, which is then used to define the subsequent opacity concepts.

Definition 1 (TA [1]). *A TA $\mathcal{A}$ is a tuple $\mathcal{A} = (\Sigma, L, \ell_0, L_{priv}, L_f, \mathbb{X}, I, E)$, where:*

1. *Σ is a finite set of actions,*
2. *L is a finite set of locations, $\ell_0 \in L$ is the initial location,*
3. *$L_{priv} \subseteq L$ is a set of private locations, $L_f \subseteq L$ is a set of final locations,*
4. *$\mathbb{X}$ is a finite set of clocks,*
5. *I is the invariant, assigning to every $\ell \in L$ a constraint $I(\ell)$ over $\mathbb{X}$ (called* invariant*),*
6. *E is a finite set of edges $e = (\ell, g, a, R, \ell')$ where $\ell, \ell' \in L$ are the source and target locations, $a \in \Sigma \cup \{\varepsilon\}$ (where ε denotes an unobservable action), $R \subseteq \mathbb{X}$ is a set of clocks to be reset, and g is a constraint over $\mathbb{X}$ (called* guard*).*

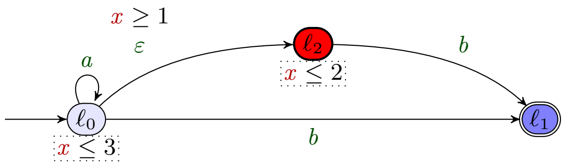


Fig. 1: A TA example

Example 1. In Fig. 1, we give an example of a TA with three locations ℓ_0, ℓ_1 and ℓ_2, three edges, two action $\{a, b\}$, and one clock x. ℓ_0 is the initial location, ℓ_2 is the (unique) private location, and ℓ_1 is the (unique) final location. ℓ_0 has an invariant "$x \leq 3$" and the edge from ℓ_0 to ℓ_2 has a guard "$x \geq 1$".

Definition 2 (Semantics of a TA). *Given a TA* $\mathcal{A} = (\Sigma, L, \ell_0, L_{priv}, L_f, \mathbb{X}, I, E)$*, the semantics of* $\mathcal{A}$ *is given by the TTS* $\mathfrak{T}_{\mathcal{A}} = (\mathfrak{S}, \mathfrak{s}_0, \Sigma \cup \{\varepsilon\} \cup \mathbb{R}_{\geq 0}, \rightarrow)$*, with*

1. $\mathfrak{S} = \big\{(\ell, \mu) \in L \times \mathbb{R}_{\geq 0}^{\mathbb{X}} \mid \mu \models I(\ell)\big\}$,
2. $\mathfrak{s}_0 = (\ell_0, \mathbf{0})$,
3. $\rightarrow \subseteq \mathfrak{S} \times E \times \mathfrak{S} \cup \mathfrak{S} \times \mathbb{R}_{\geq 0} \times \mathfrak{S}$ *consists of the discrete and (continuous) delay transition relations:*
   (a) *discrete transitions:* $((\ell, \mu), e, (\ell', \mu')) \in \rightarrow$*, and we write* $(\ell, \mu) \overset{e}{\mapsto} (\ell', \mu')$*, if* $(\ell, \mu), (\ell', \mu') \in \mathfrak{S}$*,* $e = (\ell, g, a, R, \ell') \in E$*,* $\mu' = [\mu]_R$*, and* $\mu \models g$*.*
   (b) *delay transitions:* $((\ell, \mu), d, (\ell, \mu + d)) \in \rightarrow$*, and we write* $(\ell, \mu) \overset{d}{\mapsto} (\ell, \mu + d)$*, if* $d \in \mathbb{R}_{\geq 0}$ *and* $\forall d' \in [0, d], (\ell, \mu + d') \in \mathfrak{S}$*.*

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

Given a TA $\mathcal{A}$ with semantic $(\mathfrak{S}, \mathfrak{s}_0, \Sigma \cup \{\varepsilon\} \cup \mathbb{R}_{\geq 0}, \rightarrow)$, we refer to the elements of $\mathfrak{S}$ as the *configurations* of $\mathcal{A}$. A (finite) *run* of $\mathcal{A}$ is an alternating sequence of configurations of $\mathcal{A}$ and pairs of delays and edges starting from the initial configuration $\mathfrak{s}_0$ and ending in a final configuration, of the form $(\ell_0, \mu_0), (d_0, e_0), (\ell_1, \mu_1), \cdots (\ell_{\mathrm{f}}, \mu_n)$ for some $n \in \mathbb{N}$, with $\ell_{\mathrm{f}} \in L_f$ and for $i = 0, 1, \ldots n-1$, $\ell_{\mathrm{i}} \notin L_f$, $e_i \in E$, $d_i \in \mathbb{R}_{\geq 0}$, and $(\ell_{\mathrm{i}}, \mu_i) \xrightarrow{(d_i, e_i)} (\ell_{\mathrm{i+1}}, \mu_{i+1})$. A *path* of $\mathcal{A}$ is a prefix of a run ending with a configuration.

3 Opacity problems in timed automata

3.1 Timed words, private and public runs

Given a TA $\mathcal{A}$ and a run $\rho = (\ell_0, \mu_0), (d_0, e_0), (\ell_1, \mu_1), \cdots, (\ell_{\mathrm{n}}, \mu_n)$ on $\mathcal{A}$, we say that L_{priv} is *visited in* ρ if there exists $m \in \mathbb{N}$ such that $\ell_{\mathrm{m}} \in L_{priv}$. We denote by $\mathit{Visit}^{priv}(\mathcal{A})$ the set of runs visiting L_{priv}, and refer to them as *private* runs. Conversely, we say that L_{priv} is *avoided in* ρ if the run ρ does not visit L_{priv}.

We denote the set of the runs avoiding L_{priv} by $Visit^{\overline{priv}}(\mathcal{A})$, referring to them as *public* runs.

A timed word is a sequence of pairs made of an action and an increasing timestamp in $\mathbb{R}_{\geq 0}$. We denote by $TW^*(\Sigma)$ the set of all finite timed words on the alphabet Σ. A run ρ on a TA $\mathcal{A}$ defines a timed word: if ρ is of the form $(\ell_0, \mu_0), (d_0, e_0), (\ell_1, \mu_1), \cdots, (\ell_n, \mu_n)$ where for each $i \in [\![0; n-1]\!]$, $e_i = (\ell_i, g_i, a_i, R_i, \ell_{i+1})$ and $a_i \in \Sigma \cup \{\varepsilon\}$, then it generates the timed word $(a_{j_0}, \sum_{i=0}^{j_0} d_i)(a_{j_1}, \sum_{i=0}^{j_1} d_i) \dots (a_{j_m}, \sum_{i=0}^{j_m} d_i)$, where $j_0 < j_1 < \cdots < j_m$ and $\{j_k \mid k \in [\![0; m]\!]\} = \{i \in [\![0; n-1]\!] \mid a_i \neq \varepsilon\}$. We denote by $Tr(\rho)$ and call *trace* of ρ the timed word generated by the run ρ and, by extension, given a set of runs Ω, we denote by $Tr(\Omega)$ the set of the traces of runs in Ω.

The set of timed words recognized by a TA $\mathcal{A}$ is the set of traces generated by its runs, $Tr(Visit^{priv}(\mathcal{A}) \cup Visit^{\overline{priv}}(\mathcal{A}))$ (thus a subset of $(\Sigma \times \mathbb{R}_{\geq 0})^*$). To shorten these notations, we use $Tr(\mathcal{A})$ for the set of timed words recognized by $\mathcal{A}$, also called *language* of $\mathcal{A}$. Similarly, we use $Tr^{priv}(\mathcal{A}) = Tr(Visit^{priv}(\mathcal{A}))$ to denote the set of traces of private runs, and $Tr^{\overline{priv}}(\mathcal{A}) = Tr(Visit^{\overline{priv}}(\mathcal{A}))$ for the set of traces of public runs.

In Cassez's original definition [14], actions were partitioned into two sets, depending on whether an attacker could observe them or not. For simplicity, here we replaced every unobservable transition in $\mathcal{A}$ by ε-transitions. Projecting the sequence of actions in a run onto the observable actions, as done by Cassez, is equivalent to replacing these actions by ε and taking the trace of the run. Therefore, with respect to opacity, our model is equivalent to [14].

3.2 Defining timed opacity

In this section, a definition of timed opacity based on the one from [14] is introduced, with three variants inspired by [6]: existential, full and weak opacity. If the attacker observes a set of runs of the system (i.e., observes their associated traces), we do not want them to deduce whether L_{priv} was visited or not during these observed runs. Opacity holds when the traces can be produced by both private and public runs.

We are thus first interested in the existence of an opaque trace $o \in Tr^{priv}(\mathcal{A}) \cap Tr^{\overline{priv}}(\mathcal{A})$, that is, a trace that cannot allow the attacker to decide whether it was generated by a private or a public run.

Definition 3 ($\exists$-opacity). *A TA $\mathcal{A}$ is* $\exists$-opaque *if* $Tr^{priv}(\mathcal{A}) \cap Tr^{\overline{priv}}(\mathcal{A}) \neq \emptyset$.

$\exists$-opacity decision problem:
INPUT: A TA $\mathcal{A}$
PROBLEM: Is $\mathcal{A}$ $\exists$-opaque?

Ideally and for a stronger security of the system, one can ask the system to be opaque *for all* possible traces of the system: a TA $\mathcal{A}$ is fully opaque whenever for any trace in $Tr(\mathcal{A})$, it is not possible to deduce whether the run that generated this trace visited L_{priv} or not.

Definition 4 (Full opacity). *A TA $\mathcal{A}$ is* fully opaque *if* $Tr^{priv}(\mathcal{A}) = Tr^{\overline{priv}}(\mathcal{A})$.

> **Full opacity decision problem:**
> INPUT: A TA $\mathcal{A}$
> PROBLEM: Is $\mathcal{A}$ fully opaque?

Sometimes, a weaker notion is sufficient to ensure the required security in the system, i.e., when the compromising information solely comes from the identification of the private runs.

Definition 5 (Weak opacity). *A TA $\mathcal{A}$ is* weakly opaque *if* $Tr^{priv}(\mathcal{A}) \subseteq Tr^{\overline{priv}}(\mathcal{A})$.

> **Weak opacity decision problem:**
> INPUT: A TA $\mathcal{A}$
> PROBLEM: Is $\mathcal{A}$ weakly opaque?

Example 2. The TA $\mathcal{A}$ depicted in Fig. 1 is $\exists$-opaque and weakly opaque but not fully opaque. Indeed,

$$Tr^{priv}(\mathcal{A}) = \big\{(a, \tau_1) \dots (a, \tau_n)(b, \tau_{n+1}) \mid n \in \mathbb{N} \ \wedge \ \forall i \in [\![1, n]\!], \tau_i \leq \tau_{i+1} \leq 2 \ \wedge \ \tau_{n+1} \geq 1\big\}$$

$$Tr^{\overline{priv}}(\mathcal{A}) = \big\{(a, \tau_1) \dots (a, \tau_n)(b, \tau_{n+1}) \mid n \in \mathbb{N} \wedge \forall i \in [\![1, n]\!], \tau_i \leq \tau_{i+1} \leq 3\big\}$$

This TA verifies $Tr^{priv}(\mathcal{A}) \subseteq Tr^{\overline{priv}}(\mathcal{A})$ and $Tr^{priv}(\mathcal{A}) \cap Tr^{\overline{priv}}(\mathcal{A}) \neq \emptyset$ since $(b, 1.5) \in Tr^{priv}(\mathcal{A})$.

4 Tools for the analysis of opacity

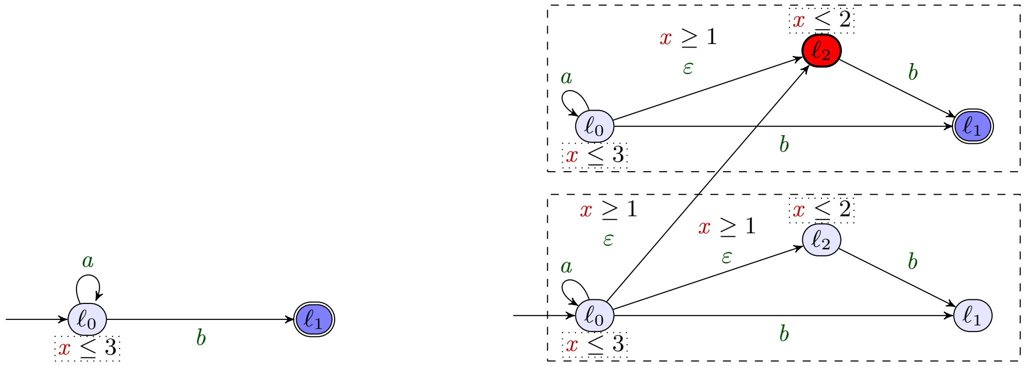
Before proving our results in Sections 5 and 6, we recall and adapt a few useful tools for TAs and opacity.

4.1 $\mathcal{A}_{priv}$ and $\mathcal{A}_{pub}$

First, we need a construction of two TAs $\mathcal{A}_{priv}$ and $\mathcal{A}_{pub}$ that recognize timed words produced respectively by private and public runs of a given TA $\mathcal{A}$.

The public runs TA $\mathcal{A}_{pub}$ is the easiest to build: it suffices to remove the private locations from $\mathcal{A}$ to eliminate every private run in the system. (See formal definition in Definition 12 in Appendix A.)

The private runs TA $\mathcal{A}_{priv}$ is obtained by duplicating all locations and transitions of $\mathcal{A}$: one copy $\mathcal{A}_S$ corresponds to the paths that already visited the private locations set, and the other copy $\mathcal{A}_{\bar{S}}$ corresponds to the paths that did not (this is a usual way to encode a Boolean, here "L_{priv} was visited", in the locations of a TA). For each private location ℓ_{priv} in $\mathcal{A}$ we copy all transitions leading to the copy of ℓ_{priv} in $\mathcal{A}_{\bar{S}}$ and redirect them to the copy of ℓ_{priv} in $\mathcal{A}_S$. The initial location is the one from $\mathcal{A}_{\bar{S}}$ and the final locations are the ones from $\mathcal{A}_S$. Hence all runs need to go from $\mathcal{A}_{\bar{S}}$ to $\mathcal{A}_S$ before reaching a final location, which requires visiting a private location.



(a) $\mathcal{A}_{pub}$ (b) $\mathcal{A}_{priv}$

Fig. 2: $\mathcal{A}_{pub}$ and $\mathcal{A}_{priv}$ with the example from Fig. 1

Definition 6 (Private runs TA $\mathcal{A}_{priv}$).

Let $\mathcal{A} = (\Sigma, L, \ell_0, L_{priv}, L_f, \mathbb{X}, I, E)$ be a TA. The private runs TA $\mathcal{A}_{priv} = (\Sigma, L_S \uplus L_{\bar{S}}, {\ell_0}^{\bar{S}}, L^S_{priv}, L^S_f, \mathbb{X}, I', E')$ is defined as follows:

1. *$L_S = \{\ell^S \mid \ell \in L\}$ and $L_{\bar{S}} = \{\ell^{\bar{S}} \mid \ell \in L\}$.*
2. *$L^S_f = \{{\ell_\mathrm{f}}^S \mid \ell_\mathrm{f} \in L_f\}$ is the set of final locations, and $L^S_{priv} = \{{\ell_{priv}}^S \mid \ell_{priv} \in L_{priv}\}$ is the set of private locations;*
3. *I' is defined such as $I'(\ell^S) = I'(\ell^{\bar{S}}) = I(\ell)$*
4. *$E' = E_S \uplus E_{\bar{S}} \uplus E_{\bar{S}\to S}$ where E_S and $E_{\bar{S}}$ are the two disjoint copies of E respectively associated to the sets of locations L_S and $L_{\bar{S}}$, and $E_{\bar{S}\to S}$ is a copy of the set of all transitions that go toward $L^{\bar{S}}_{priv}$ where the target location ${\ell_{priv}}^{\bar{S}}$ has been changed into ${\ell_{priv}}^S$. More formally:*

$$\begin{aligned} E_S &= \big\{(\ell^S, g, a, R, \ell'^S) \mid (\ell, g, a, R, \ell') \in E\big\} \\ E_{\bar{S}} &= \big\{(\ell^{\bar{S}}, g, a, R, \ell'^{\bar{S}}) \mid (\ell, g, a, R, \ell') \in E\big\} \\ E_{\bar{S}\to S} &= \big\{(\ell^{\bar{S}}, g, a, R, {\ell_{priv}}^S) \mid (\ell, g, a, R, \ell_{priv}) \in E\big\}. \end{aligned}$$

Example 3. We illustrate these constructions in Fig. 2 with $\mathcal{A}$ from Fig. 1.

The languages of $\mathcal{A}_{priv}$ and $\mathcal{A}_{pub}$ are respectively $Tr^{priv}(\mathcal{A})$ and $Tr^{\overline{priv}}(\mathcal{A})$.

Remark 1. By a minor modification on $\mathcal{A}_{priv}$, one can build a TA $\mathcal{A}_{mem}$ that recognizes exactly the same language as $\mathcal{A}$ and that stores in each location whether the private locations set has been visited. To do so, we add the set $\{{\ell_\mathrm{f}}^{\bar{S}} \mid \ell_\mathrm{f} \in L_f\}$ to the set of final locations in $\mathcal{A}_{priv}$ and we remove each ${\ell_{priv}}^{\bar{S}} \in L^{\bar{S}}_{priv}$ from $L_{\bar{S}}$ in the same way as we did in $\mathcal{A}_{pub}$. Notably, $\mathcal{A}$ is weakly (resp. fully) opaque if and only if $\mathcal{A}_{mem}$ is weakly (resp. fully) opaque.

4.2 Inter-reducibility of weak and full opacity

While the distinction between weak and full notions of opacity can lead to meaningful changes [6], within our framework both associated problems are inter-reducible.

Theorem 1. *The weak opacity decision problem and the full opacity decision problem are inter-reducible.*

Proof. Let us first show that the full opacity decision problem reduces to the weak opacity decision problem. Let $\mathcal{A}$ be a TA. In order to test whether $\mathcal{A}$ is fully opaque, we can test both inclusions: $Tr^{priv}(\mathcal{A}) \subseteq Tr^{\overline{priv}}(\mathcal{A})$ and $Tr^{priv}(\mathcal{A}) \supseteq Tr^{\overline{priv}}(\mathcal{A})$. The first inclusion can be decided directly by testing whether $\mathcal{A}$ is weakly opaque. In order to test the second inclusion, we need to build a TA $\mathcal{A}_{mem}$ where private and public runs are inverted. To do so, we first build $\mathcal{A}_{pub}$ and $\mathcal{A}_{priv}$ and then define $\mathcal{A}'$ as the TA constituted of $\mathcal{A}_{pub}$ and $\mathcal{A}_{priv}$ as well as two new locations ${\ell_0}'$ and ${\ell_{priv}}'$. The location ${\ell_0}'$ is the initial location of $\mathcal{A}'$ and ${\ell_{priv}}'$ is the only private location. For $x \in \mathbb{X}$, both ${\ell_0}'$ and ${\ell_{priv}}'$ have the invariant $x = 0$, ensuring no time may elapse in those locations. From ${\ell_0}'$, with a transition labeled by ε, one may reach either the initial location of $\mathcal{A}_{priv}$ (${\ell_0}^S$) or ${\ell_{priv}}'$, from which an ε-transition leads to the initial location of $\mathcal{A}_{pub}$ (ℓ_0). The final locations of $\mathcal{A}'$ are the final locations of $\mathcal{A}_{pub}$ and $\mathcal{A}_{priv}$. The public runs of $\mathcal{A}'$ are the ones starting in ${\ell_0}'$, going immediately to ${\ell_0}^S$, and then following a run of $\mathcal{A}_{priv}$ until a final location of $\mathcal{A}_{priv}$ is reached. As the initial transition is labeled by ε, we have $Tr^{\overline{priv}}(\mathcal{A}') = Tr^{priv}(\mathcal{A})$. Similarly, the private runs of $\mathcal{A}'$ are the ones starting in ${\ell_0}'$, going immediately to ${\ell_{priv}}'$ followed immediately by going to ${\ell_0}^S$, and then follows a run of $\mathcal{A}_{pub}$ until a final location of $\mathcal{A}_{pub}$ is reached. As the two initial transitions are labeled by ε, we have $Tr^{priv}(\mathcal{A}') = Tr^{\overline{priv}}(\mathcal{A})$. Hence, $\mathcal{A}$ is fully opaque if and only if $\mathcal{A}$ and $\mathcal{A}'$ are weakly opaque.

Let us now show the converse reduction. Let $\mathcal{A}$ be a TA. We will define a TA $\mathcal{A}'$ such that $\mathcal{A}'$ is fully opaque if and only if $\mathcal{A}$ is weakly opaque. To do so, we want that $Tr^{\overline{priv}}(\mathcal{A}') = Tr^{\overline{priv}}(\mathcal{A})$ and $Tr^{priv}(\mathcal{A}') = Tr^{\overline{priv}}(\mathcal{A}) \cup Tr^{priv}(\mathcal{A})$. Indeed, if these equalities hold, $Tr^{priv}(\mathcal{A}') = Tr^{\overline{priv}}(\mathcal{A}')$ would be equivalent to $Tr^{\overline{priv}}(\mathcal{A}) = Tr^{\overline{priv}}(\mathcal{A}) \cup Tr^{priv}(\mathcal{A})$ which holds if and only if $Tr^{priv}(\mathcal{A}) \subseteq Tr^{\overline{priv}}(\mathcal{A})$. As for the first reduction, $\mathcal{A}'$ contains a copy of $\mathcal{A}_{pub}$ and $\mathcal{A}_{priv}$ as well as two new locations ${\ell_0}'$ and ${\ell_{priv}}'$. The location ${\ell_0}'$ is the initial location of $\mathcal{A}'$ and ${\ell_{priv}}'$ is the only private location. For $x \in \mathbb{X}$, both ${\ell_0}'$ and ${\ell_{priv}}'$ have the invariant $x = 0$, ensuring no time may elapse in those locations. From ${\ell_0}'$, with a transition labeled by ε, one may reach either the initial location of $\mathcal{A}_{pub}$ (${\ell_0}^S$) or ${\ell_{priv}}'$, from which an ε-transition leads either to ${\ell_0}^S$ or to the initial location of $\mathcal{A}_{pub}$ (ℓ_0). The final locations of $\mathcal{A}'$ are the final locations of $\mathcal{A}_{pub}$ and $\mathcal{A}_{priv}$. The public runs of $\mathcal{A}'$ are the ones starting in ${\ell_0}'$, going immediately to ℓ_0, and then following a run of $\mathcal{A}_{pub}$ until a final location of $\mathcal{A}_{pub}$ is reached. As the initial transition is labeled by ε, we have $Tr^{\overline{priv}}(\mathcal{A}') = Tr^{\overline{priv}}(\mathcal{A})$. Similarly, the private runs of $\mathcal{A}'$ are the ones starting in ${\ell_0}'$, going immediately to ${\ell_{priv}}'$ followed immediately by going to ${\ell_0}^S$ followed by a run of $\mathcal{A}_{priv}$, or to ℓ_0, followed by a run of $\mathcal{A}_{pub}$ until a final location of$\mathcal{A}_{pub}$ is reached. As the two initial transitions are labeled by ε, we have $Tr^{priv}(\mathcal{A}') = Tr^{priv}(\mathcal{A}) \cup Tr^{\overline{priv}}(\mathcal{A})$. Hence, $\mathcal{A}$ is weakly opaque if and only if $\mathcal{A}'$ is fully opaque.

4.3 Region automaton

We recall that the region automaton is obtained by quotienting the set of clock valuations out by an equivalence relation $\simeq$ recalled below.

Given a TA $\mathcal{A}$ and its set of clocks $\mathbb{X}$, we define $M : \mathbb{X} \to \mathbb{N}$ the map that associates to a clock x the greatest value to which the interpretations of x are compared within the guards and invariants; if x appears in no constraint, we set $M(x) = 0$.

Given $\alpha \in \mathbb{R}$, we write $\lfloor \alpha \rfloor$ and $\mathrm{fr}(\alpha)$ respectively for the integral and fractional parts of α.

Definition 7 (Equivalence relation $\simeq$ for valuations [1]). *Let μ, μ' be two clock valuations (with values in $\mathbb{R}_{\geq 0}$). We say that μ and μ' are equivalent, denoted by $\mu \simeq \mu'$ when, for each $x \in \mathbb{X}$, either $\mu(x) > M(x)$ and $\mu'(x) > M(x)$ or the three following conditions hold:*

1. *$\lfloor \mu(x) \rfloor = \lfloor \mu'(x) \rfloor$;*
2. *$fr(\mu(x)) = 0$ if and only if $fr(\mu'(x)) = 0$;*
3. *for each $y \in \mathbb{X}$, $fr(\mu(x)) \leq fr(\mu(y))$ if and only if $fr(\mu'(x)) \leq fr(\mu'(y))$.*

The equivalence relation is extended to the configurations of $\mathcal{A}$: let $\mathfrak{s} = (\ell, \mu)$ and $\mathfrak{s}' = (\ell', \mu')$ be two configurations in $\mathcal{A}$, then $\mathfrak{s} \simeq \mathfrak{s}'$ if and only if $\ell = \ell'$ and $\mu \simeq \mu'$.

The equivalence class of a valuation μ is denoted $[\mu]$ and is called a *clock region*, and the equivalence class of a configuration $\mathfrak{s} = (\ell, \mu)$ is denoted $[\mathfrak{s}]$ and called a *region* of $\mathcal{A}$. Clock regions are denoted by the enumeration of the constraints defining the equivalence class. Thus, values of a clock x that go beyond $M(x)$ are merged and described in the regions by "$x > M(x)$".

The set of regions of $\mathcal{A}$ is denoted by $\mathcal{R}_\mathcal{A}$. These regions are in finite number: this allows us to construct a finite "untimed" regular automaton, the region automaton $\mathcal{RA}_\mathcal{A}$. Locations of $\mathcal{RA}_\mathcal{A}$ are regions of $\mathcal{A}$, and the transitions of $\mathcal{RA}_\mathcal{A}$ convey the reachable valuations associated to each configuration in $\mathcal{A}$.

To formalize the construction, we need to transform discrete and time-elapsing transitions of $\mathcal{A}$ into transitions between the regions of $\mathcal{A}$. To do that, we define a "time-successor" relation that corresponds to time-elapsing transitions.

Definition 8 (Time-successor relation [7]). *Let $r = (\ell, [\mu]), r' = (\ell', [\mu']) \in \mathcal{R}_\mathcal{A}$. We say that r' is a time-successor of r when $r \neq r'$, $\ell = \ell'$ and for each configuration (ℓ, μ) in r, there exists $d \in \mathbb{R}_{\geq 0}$ such that $(\ell, \mu + d)$ is in r' and for all $d' < d$, $(\ell, \mu + d') \in r \cup r'$.*

A region $r = (\ell, [\mu])$ is *unbounded* when, for all x in $\mathbb{X}$, $\mu(x) > M(x)$.

Definition 9 (Region automaton [1]). *Given a TA* $\mathcal{A} = (\Sigma, L, \ell_0, L_{priv}, L_f, \mathbb{X}, I, E)$, *the region automaton is a tuple* $\mathcal{RA}_{\mathcal{A}} = (\Sigma_{\mathcal{R}}, \mathcal{R}, r_0, \mathcal{R}_f, E_{\mathcal{R}})$ *where*

1. $\Sigma_{\mathcal{R}} = \Sigma \cup \{\varepsilon\}$;
2. $\mathcal{R} = \mathcal{R}_{\mathcal{A}}$;
3. $r_0 = [\mathfrak{s}_0]$;
4. $\mathcal{R}_f$ *is the set of regions which first component is a final location* $\ell_f \in L_f$;
5. (discrete transitions) *For every* $r = (\ell, [\mu])$ *with* $\ell \notin L_f$, $r' = (\ell', [\mu']) \in \mathcal{R}_{\mathcal{A}}$ *and* $a \in \Sigma \cup \{\varepsilon\}$:

$$(r, a, r') \in E_{\mathcal{R}} \text{ if } (\ell, \mu) \overset{e}{\mapsto} (\ell', \mu'')$$

with $\mu'' \in [\mu']$ *and* $e = (\ell, g, a, R, \ell') \in E$.
(delay transitions) *For every* $r = (\ell, [\mu])$ *with* $\ell \notin L_f$, $r' = (\ell', [\mu']) \in \mathcal{R}_{\mathcal{A}}$:

$$(r, \varepsilon, r') \in E_{\mathcal{R}} \text{ if } r' \text{ is a time-successor of } r \text{ or if } r = r' \text{ is unbounded.}$$

As in TAs, a *run* of $\mathcal{RA}_{\mathcal{A}}$ is an alternating sequence of regions of $\mathcal{RA}_{\mathcal{A}}$ and actions starting from the initial region r_0 and ending in a final region, of the form $r_0, a_0, r_1, a_1, \cdots r_{n-1}, a_{n-1}, r_f$ for some $n \in \mathbb{N}$, with $r_f \in R_f$ and for $i \in [\![0; n-1]\!]$, $r_i \notin R_f$, and $(r_i, a_i, r_{i+1}) \in E_{\mathcal{R}}$. A *path* of $\mathcal{RA}_{\mathcal{A}}$ is a prefix of a run ending with a region and the trace of a path of $\mathcal{RA}_{\mathcal{A}}$ is the sequence of actions (ε excluded) contained in this path.

5 Opacity problems for subclasses of timed automata

In this section, we consider the decidability status of the three opacity problems presented in Section 3 for several subclasses of TAs: TAs with one clock, TAs with one action, TAs under discrete time and observable ERAs. We first show the decidability of the $\exists$-opacity problem in the general case. Then, we focus on each class of TAs listed above to study weak and full opacity.

5.1 $\exists$-opacity problem

We prove here decidability of the $\exists$-opacity problem in the general case. We establish in the following proof a reduction from the $\exists$-opacity problem to the reachability problem in TAs, which is known to be in PSPACE [1].

Theorem 2 (Decidability of $\exists$-opacity). $\exists$*-opacity is decidable for TAs.*

Proof. Let $\mathcal{A}$ be a TA. We build $\mathcal{A}_{priv}$ and $\mathcal{A}_{pub}$ from $\mathcal{A}$ as described in Section 4.1. Noting that the product of two TAs recognizes the intersection of their languages [1, Theorem 3.15] (assuming the two TAs share no clock), we build the TA $\mathcal{A}_{priv} \times \mathcal{A}_{pub}$, product of $\mathcal{A}_{priv}$ and $\mathcal{A}_{pub}$, which language is

$Tr^{priv}(\mathcal{A}) \cap Tr^{\overline{priv}}(\mathcal{A})$. To build this product, we can rename all clocks from $\mathcal{A}_{pub}$ so that $\mathcal{A}_{priv}$ and $\mathcal{A}_{pub}$ share no clock.

The $\exists$-opacity problem is by definition the non-emptiness of the intersection of $Tr^{priv}(\mathcal{A})$ and $Tr^{\overline{priv}}(\mathcal{A})$. Moreover, the reachability of a final location of $\mathcal{A}_{priv} \times \mathcal{A}_{pub}$ is equivalent to the non-emptiness of the language of $\mathcal{A}_{priv} \times \mathcal{A}_{pub}$, and thus of the set $Tr^{priv}(\mathcal{A}) \cap Tr^{\overline{priv}}(\mathcal{A})$. Since reachability is decidable in PSPACE in TAs [1], the same holds for the $\exists$-opacity problem.

5.2 Timed automata with a single action

Recall that the universality problem consists in deciding whether a TA $\mathcal{A}$ accepts the set of all timed words. In [18], it is shown that the class of one-action TAs is one of the simplest cases for which the universality problem is undecidable among TAs. Therefore, this gives the intuition that the weak opacity and full opacity problems are undecidable as well for one-action TAs ($|\Sigma| = 1$). We establish this intuition in this section.

Theorem 3 (Undecidability of universality in one-action TAs [18]). *The problem of universality for TAs with one action is undecidable.*

We present in the proof of the following theorem a reduction of the universality problem in one-clock TAs to the full opacity problem in the same context.

Theorem 4 (Undecidability of full opacity in one-action TAs). *The full opacity problem for TAs with one action is undecidable.*

Proof. Let $\mathcal{A}$ be a TA with a single action. We want to build a TA such that if we can answer the full opacity problem of this TA, then we can decide the universality problem for $\mathcal{A}$. We consider the following TA: we add an initial location exited by two ε-transitions that must be taken urgently (i.e., no time may elapse before taking them). The first ε-transition leads to a secret location which leads (again via an urgent ε-transition) to the initial location of the TA $\mathcal{A}$ and the other leads to a location where every finite timed words on Σ can be read before reaching a final location. We denote this TA $\mathcal{B}$ and illustrate its construction in Fig. 3. The language recognized by $\mathcal{A}$ corresponds exactly to the traces of private runs of $\mathcal{B}$, and the traces of public runs of $\mathcal{B}$ are all the finite timed words on Σ. Therefore, $\mathcal{B}$ is fully opaque iff $Tr^{priv}(\mathcal{B}) = Tr^{\overline{priv}}(\mathcal{B})$ iff $Tr(\mathcal{A}) = TW^*(\Sigma)$ iff $\mathcal{A}$ is universal. By Theorem 3, we conclude that the full opacity problem for one-action TAs is undecidable.

With Theorem 1, we deduce:

Corollary 1 (Undecidability of weak opacity in one-action TAs). *The weak opacity problem for TAs with one action is undecidable.*

Remark 2. The problems of execution-time opacity introduced in [6] are a particular *decidable* subcase of these undecidable opacity problems with one-action TAs. Indeed, the execution time is equivalent to a *unique* timestamp associated to the last action of the system.

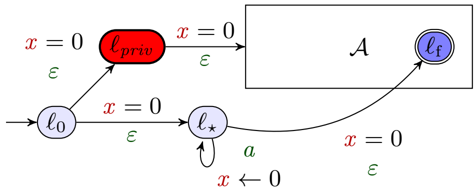


Fig. 3: Automaton $\mathcal{B}$: Reduction from universality to full opacity

In addition, due to the undecidability of language universality for TAs with at least two clocks [18, Theorem 21], we can prove the following with the same construction as in Theorem 4:

Theorem 5 (Undecidability of opacity for two-clock TAs). *Full and weak opacity are undecidable for TAs with ≥ 2 clocks.*

5.3 Timed automata with a single clock

We now prove that the weak opacity and full opacity problems are both decidable in the context of one-clock TAs ($|\mathbb{X}| = 1$) relying on the following result from [18].

Theorem 6 (Decidability of language inclusion in one-clock TAs [18]). *The language inclusion problem for one-clock TAs is decidable.*

By definition, a TA is weakly opaque if $Tr^{priv}(\mathcal{A})$ is included in $Tr^{\overline{priv}}(\mathcal{A})$. As $Tr^{priv}(\mathcal{A})$ and $Tr^{\overline{priv}}(\mathcal{A})$ are respectively recognized by $\mathcal{A}_{priv}$ and $\mathcal{A}_{pub}$, the decidability of the weak opacity problem is directly obtained from the decidability of the inclusion of two languages.

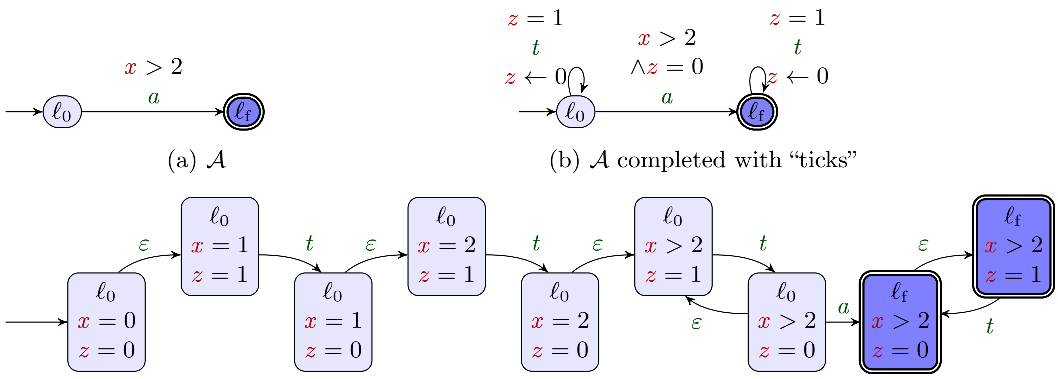
Theorem 7 (Decidability of weak opacity in one-clock TAs). *Weak opacity is decidable for one-clock TAs.*

With Theorem 1, we deduce:

Corollary 2 (Decidability of full opacity in one-clock TAs). *Full opacity is decidable for one-clock TAs.*

5.4 Timed automata over discrete time

In the general case, clocks are real-valued variables, with valuations thus ranging over $\mathbb{T} = \mathbb{R}_{\geq 0}$. TAs over discrete time however restrict the clock's behavior to valuations over $\mathbb{T} = \mathbb{N}$. Since the arguments used in [1] to prove the undecidability of the universality problem in TAs rely on the continuous time, this proof cannot be used to establish undecidability of opacity over discrete time. In fact, relying on the region automaton (defined in Section 4.3) in discrete time and classical results on finite regular automata, we show *decidability* of the opacity problems.



(a) $\mathcal{A}$

(b) $\mathcal{A}$ completed with "ticks"

(c) $\mathcal{RA}_\mathcal{A}$

Fig. 4: A discrete-time region automaton example

If μ, μ' are two discrete clock valuations (i.e., with values in $\mathbb{N}$), the definition of $\simeq$ from Section 4.3 can be simplified into: $\mu \simeq \mu'$ if and only if for each $x \in \mathbb{X}$, either $\mu(x) = \mu'(x)$ or $\mu(x) > M(x)$ and $\mu'(x) > M(x)$.

Over continuous time, for each run of the TA, there is a unique corresponding run of the region automaton. Over discrete time, thanks to the simplified form of the definition of $\simeq$, the converse statement that a run of the region automaton corresponds to a unique run of the TA nearly holds. Loss of information however remains when every clock goes beyond their maximum constant, as time elapsing is not measured beyond this point. In order to measure it, we add a letter t for "tick" which occurs each time that an (integral) time unit passes in the region automaton. This change can be operated directly on the TA $\mathcal{A}$ so that the correspondence between paths of $\mathcal{A}$ and $\mathcal{RA}_\mathcal{A}$ becomes immediate.

More precisely, we add a clock z and add self-loop transitions $e_t = (\ell, (z = 1), t, \{z\}, \ell)$ on each location $\ell \in L$ of $\mathcal{A}$. We also add the guard "$z = 0$" to each discrete transition of $\mathcal{A}$.

We illustrate the resulting TA on a simple example in Fig. 4. We depict a discrete-time TA $\mathcal{A}$, its transformation by the procedure we just described and finally its region automaton $\mathcal{RA}_\mathcal{A}$ (over discrete time).

With this construction, time information become superfluous in the TA as it can be deduced from the number of ticks that were produced, which also appears within a path of the region automaton. For instance, consider the run on the $\mathcal{A}$ of Fig. 4a that remains four time units in ℓ_0 before going to ℓ_f. The timed word $(a, 4)$ on the original TA $\mathcal{A}$ becomes $(t, 1)(t, 2)(t, 3)(t, 4)(a, 4)$ in our transformed TA. The untimed word obtained in $\mathcal{RA}_\mathcal{A}$ is $tttta$, which means that four "ticks" occurred before the action a was produced. From this information, the original timed word $(a, 4)$ can be reconstructed. In the rest of this subsection, we only consider TAs enhanced with ticks. From the previous discussion, we have:

Lemma 1. *The language of a discrete-time TA and the language of its region automaton are in bijection.*

Thus, we show that the language inclusion problem for discrete-time TAs can be reduced to its decidable equivalent for finite regular automata. This result is not surprising—yet, we could not find its occurrence in the literature.

Proposition 1 (Decidability of language inclusion in discrete-time TAs). *Language inclusion in discrete-time TAs is decidable.*

We can then adapt this result to the weak and full opacity problems in a similar way as done in Section 5.3.

Theorem 8 (Decidability of weak and full opacity in discrete-time TAs). *Weak and full opacity are decidable for discrete-time TAs.*

5.5 Observable Event-Recording Automata

In [14], the opacity problems were shown to be undecidable for Event-Recording Automata (ERAs) [2], a subclass of TAs where every clock x is associated to a specific event a_x and x is reset on a transition iff this transition is labeled by a_x. Due to this, the valuations of clocks are entirely determined by the duration since the last occurrence of the associated events. One of the main interest of ERAs is that they are determinizable [2]. This determinization is carried out through the standard subset construction.

The undecidability result from [14] on ERAs required to make the events a_x unobservable. Hence, in our framework they would be replaced by ε-transitions. We define observable ERAs (oERAs) as ERAs where the actions resetting the clocks must be observable. This means that the information required for the determinization now belongs to the trace that is observed.

Given an oERA $\mathcal{A}$, we can thus build through the subset construction a TA $Det_{\mathcal{A}}$ such that any path ρ in $\mathcal{A}$ corresponds to a path ρ_D in $Det_{\mathcal{A}}$ with the same trace and ending in a location labeled by the set of all the locations of $\mathcal{A}$ that can be reached with a run that has the same trace as ρ. This information, combined with the construction of $\mathcal{A}_{mem}$ (Remark 1) which stores in the state of the TA whether the private location was visited or not, directly provides the following result.

Theorem 9 (Decidability of the opacity problems in oERAs). *Weak and full opacity are decidable for oERAs.*

6 Opacity after a finite number of observations

One of the causes for the undecidability of the opacity problems in [14] stems from the unbounded memory the attacker might require to remember a run of the TA. As a consequence, one can wonder whether the opacity problems remain

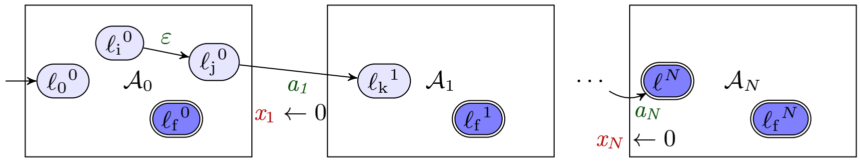


Fig. 5: Partially unfolded TA

undecidable when the attacker performs only a *finite* number of observations. In this section, we prove that the weak and full opacity problems become decidable whenever, given $N \in \mathbb{N}$, the attacker only observes the first N actions (with their timestamps).

For instance, if $(a, 1.2)(b, 1.4)(b, 1.5)(a, 2.1)$ is the trace of a public run of the system, and $N = 2$, then the attacker only observes the trace $(a, 1.2)(b, 1.4)$. If $(a, 1.2)(b, 1.4)(c, 1.6)$ is the trace of a private run, the trace observed by the attacker is $(a, 1.2)(b, 1.4)$ again and the attacker cannot conclude a private run occurred or not.

Formally, and in order to define new variants of opacity representing this framework, given a TA $\mathcal{A}$, we define a new TA (depicted in Fig. 5) which emulates the behavior of $\mathcal{A}$ up to the Nth observation. This TA is an unfolding of $\mathcal{A}$ with $N+1$ copies of $\mathcal{A}$, where ε-transitions are taken within each copy, and transitions with an observable action lead to the next copy. A run ends when either a final location or the final copy is reached.

Additionally, and in order to prepare our proof, we directly enrich this TA with ticks. In Section 5.4, we added a single tick to the automaton which counted the time elapsed since the start of the run. Here, the TA includes as well, for each $i < N$, a tick counting the time elapsed since the ith observation. As multiple ticks may need to occur at the same time, we develop the alphabet of ticks to describe the set of tick clocks that need to be reset, i.e., the tick $t_{\{i_1,\dots,i_m\}}$ is produced by the TA if for every j, the i_jth observation (or the start of the run if $i_j = 0$) occurred an integer number of time units beforehand. Note that the addition of these ticks immediately uses the assumption that only N actions are observed.

Definition 10 (N-observations unfolding of a TA).

Let $\mathcal{A} = (\Sigma, L, \ell_0, L_{priv}, L_f, \mathbb{X}, I, E)$ be a TA and let $N \in \mathbb{N}$. We call N-unfolding of $\mathcal{A}$ the TA $\mathit{Unfold}_N(\mathcal{A}) = (\Sigma', L', {\ell_0}^0, L'_{priv}, L'_f, \mathbb{X}', I', E')$ where

1. *$\Sigma' = \Sigma \cup \Sigma_0 \cup \Sigma_t$ where $\Sigma^0 = \{a_0 \mid a \in \Sigma\}$ is a copy of the alphabet Σ that is used to represent within the action's name that it occurred at the same time as the previous action, and $\Sigma_t = \{t_K \mid K \subseteq [\![0; N]\!]\}$ is the set of "ticks" associated to each set of added clocks;*
2. *$L' = \bigcup_{i=0}^{N} L^i$ where the sets L^i are $N+1$ disjoint copies of L where each location $\ell \in L$ has been renamed $\ell^i \in L^i$: for $0 \leq i \leq N$, $L^i = \{\ell^i \mid \ell \in L\}$;*
3. *${\ell_0}^0 \in L^0$ is the initial location;*

4. $L'_{priv} = \bigcup_{i=0}^{N-1} L^i_{priv}$ *where* L^i_{priv} *are the copies within* L^i *of the private locations of* $\mathcal{A}$*;*
5. $L'_f = (\bigcup_{i=0}^{N} L^i_f) \cup L^N$ *where* L^i_{priv} *are the copies within* L^i *of the final locations of* $\mathcal{A}$*;*
6. $\mathbb{X}' = \mathbb{X} \cup \{x_i \mid i \in [\![0;N]\!]\}$*, the original clocks to which* $N+1$ *clocks* $(x_0, \dots, x_{N+1})$ *were added for ticks;*
7. $I'(\ell^i) = I(\ell)$ *for* $l \in L$ *and* $i \leq N$ *extends* I *to each* L_i*;*
8. $E' = \bigcup_{i=0}^{N-1} E^i \cup E^{i \to i+1}$ *is the set of transitions where, given* $0 \leq i < N$
   - $E^i = \{(\ell^i, \varepsilon, g \wedge \bigwedge_{k=0}^{i} (x_k < 1), R, \ell'^i) \mid (\ell, \varepsilon, g, R, \ell') \in E\} \cup \{(\ell^i, t_K, \bigwedge_{k \in K} (x_k = 1) \wedge \bigwedge_{m \in [\![0;i]\!] \setminus K} (0 < x_m < 1), \{x_k \mid k \in K\}, \ell^i) \mid \ell^i \in L^i \wedge K \subseteq [\![0;i]\!]\}$;
   - $E^{i \to i+1} = \{(\ell^i, a^0, g \wedge \bigwedge_{k=0}^{i} (x_k < 1) \wedge \bigvee_{m=0}^{i} (x_m = 0), R \cup \{x_{i+1}\}, \ell'^{i+1}) \mid (\ell, a, g, R, \ell') \in E \wedge a \in \Sigma\} \cup \{(\ell^i, a, g \wedge \bigwedge_{k=0}^{i} (0 < x_k < 1), R \cup \{x_{i+1}\}, \ell'^{i+1}) \mid (\ell, a, g, R, \ell') \in E \wedge a \in \Sigma\}$.

Definition 11 (Opacity w.r.t. N observations). *Let* $\mathcal{A}$ *be a TA and let* $N \in \mathbb{N}$*. We say that* $\mathcal{A}$ *is* weakly (resp. fully) opaque w.r.t. N observations *when* $\mathit{Unfold}_N(\mathcal{A})$ *is weakly (resp. fully) opaque.*

Theorem 10. *The problem of deciding, given a TA* $\mathcal{A}$ *and* $N \in \mathbb{N}$*, whether* $\mathcal{A}$ *is weakly (resp. fully) opaque w.r.t.* N *observations is decidable.*

The rest of the this section is devoted to the proof of Theorem 10.

As in Section 5.4, the goal is to rely on the region automaton to translate the opacity problems from the TA to another problem on a finite automaton. Let $\mathcal{A} = (\Sigma, L, \ell_0, L_{priv}, L_f, \mathbb{X}, I, E)$ be a TA and let $N \in \mathbb{N}$. Before unfolding $\mathcal{A}$, we replace it by the TA $\mathcal{A}_{mem}$ described in Remark 1. Recall that $\mathcal{A}_{mem}$ recognizes the same language as $\mathcal{A}$ but stores within the locations the information whether L_{priv} was visited. As such, $\mathcal{A}_{mem}$ has the same opacity properties as $\mathcal{A}$, so we can consider $\mathit{Unfold}_N(\mathcal{A}_{mem})$ instead of $\mathit{Unfold}_N(\mathcal{A})$ to study the opacity of $\mathcal{A}$.

Let $\mathcal{RA}_{\mathit{Unfold}_N(\mathcal{A}_{mem})}$ be the region automaton of $\mathit{Unfold}_N(\mathcal{A}_{mem})$. Thanks to the added ticks, paths of $\mathcal{RA}_{\mathit{Unfold}_N(\mathcal{A}_{mem})}$ sharing the same trace correspond to runs of $\mathcal{A}$ for which the (at most) N observations occurred within the same time intervals (due to the tick representing the total time) and the fractional part of the timing of those observations have the same order. This is the information we mainly need, and thus we wish to regroup every path of the region automaton with the same trace. As the region automaton is a finite automaton, we can realize usual operations on it, that is, first remove ε-transitions (by fusing them with the following non-ε-transition) and then determinizing the automaton through the subset construction. We denote by $\mathcal{B}(\mathcal{A})$ the resulting automaton. We call *beliefs* the states of $\mathcal{B}(\mathcal{A})$, i.e., they describe the set of regions the attacker believes the system may be in.

Let B be a belief of $\mathcal{B}(\mathcal{A})$ and B_{priv} (resp. B_{pub}) be the subset of B containing the regions which associated location in $\mathcal{A}_{mem}$ is private (resp. public) and final. We say that B is weakly (resp. fully) *discriminating* if $B_{priv} \neq \emptyset$ and $B_{pub} = \emptyset$ (resp. if either $B_{priv} \neq \emptyset$ and $B_{pub} = \emptyset$ or $B_{priv} = \emptyset$ and $B_{pub} \neq \emptyset$). The discriminating belief in $\mathcal{B}(\mathcal{A})$ allows to characterize the opacity problems.

Proposition 2 (Relation between opacity and discriminating belief). *A TA $\mathcal{A}$ is weakly (resp. fully) opaque w.r.t. N observations iff $\mathcal{B}(\mathcal{A})$ does not contain any weakly (resp. fully) discriminating belief.*

Proof. We focus on weak opacity, full opacity case can be treated similarly.

- Assume first that $\mathcal{B}(\mathcal{A})$ contains a weakly discriminating belief B. Let r be a region in B_{priv} and w be the trace of a path leading from the initial belief of $\mathcal{B}(\mathcal{A})$ to B. By construction of the region automaton, there exists a run ρ of $\mathit{Unfold}_N(\mathcal{A}_{mem})$ whose untimed trace (i.e., the trace of ρ projected on the actions) is w and such that the run corresponding to ρ in the region automaton ends in r. In particular, ρ is a private run. Moreover, any run whose untimed trace is w ends in a region of B. Thus, there is no public run with trace w and in particular $Tr(\rho) \in Tr^{priv}(\mathit{Unfold}_N(\mathcal{A}_{mem}))$ and $Tr(\rho) \in Tr^{\overline{priv}}(\mathit{Unfold}_N(\mathcal{A}_{mem}))$, hence $\mathit{Unfold}_N(\mathcal{A}_{mem})$ is not weakly opaque and $\mathcal{A}$ is not weakly opaque w.r.t. N observations.
- Assume now that $\mathcal{A}$ is not weakly opaque w.r.t. N observations. Let ρ be a run of $\mathit{Unfold}_N(\mathcal{A}_{mem})$ such that $Tr(\rho) \in Tr^{priv}(\mathit{Unfold}_N(\mathcal{A}_{mem}))$ and $Tr(\rho) \notin Tr^{\overline{priv}}(\mathit{Unfold}_N(\mathcal{A}_{mem}))$. Let $[\rho]$ be the run corresponding to ρ in the region automaton.[1] We denote by $T([\rho])$ the set of traces of runs of $\mathit{Unfold}_N(\mathcal{A}_{mem})$ associated to $[\rho]$.

  Lemma 2. *Denoting $w = a_1, \dots a_m$ the trace of $[\rho]$, $T([\rho])$ contains exactly the words $(a_1, \tau_1) \dots (a_m, \tau_m)$ satisfying the following constraints:*
  1. $\forall i \in \llbracket 1; m \rrbracket, (a_i \in \Sigma \cup \Sigma_t \implies \tau_i - \tau_{i-1} > 0) \wedge (a_i \in \Sigma_0 \implies \tau_i - \tau_{i-1} = 0)$ *(where $\tau_0 = 0$), meaning that only the actions of Σ_0 can be made without any delay.*
  2. $\forall i, j \in \llbracket 0; m \rrbracket, \forall k \in \llbracket i+1; j-1 \rrbracket, \forall J, K \subseteq \llbracket 0; N \rrbracket, \forall I \subseteq J, (i < j \wedge a_i = t_I \wedge a_j = t_J \wedge (a_k = t_K \implies K \cap J = \emptyset)) \implies \tau_j - \tau_i = 1$*, meaning that two successive ticks of the same clocks are separated by exactly 1 time unit.*
  3. $\exists i \in \llbracket 1; m \rrbracket, \exists I \subseteq \llbracket 0; N \rrbracket \forall j \leq i, \forall J \subseteq \llbracket 0; N \rrbracket, a_i = t_I \wedge 0 \in I \wedge \tau_i = 1 \wedge (a_j = T_J \implies 0 \notin J)$*, meaning that the first occurrence of the tick of the clock x_0 is at time 1.*

[1] The notation $[\cdot]$ represents that $[\rho]$ implicitly defines an equivalence class of runs of $\mathit{Unfold}_N(\mathcal{A}_{mem})$. For a run ρ' of $\mathit{Unfold}_N(\mathcal{A}_{mem})$, we thus write $\rho' \in [\rho]$ to say that the run associated to ρ' in the region automaton is $[\rho]$.

Subclass	∃-opacity	weak opacity	full opacity
$\lvert\mathbb{X}\rvert = 1$	$\surd$Theorem 2	$\surd$Theorem 7	$\surd$Corollary 2
$\lvert\mathbb{X}\rvert = 2$	$\surd$Theorem 2	$\times$Theorem 5	
$\lvert\Sigma\rvert = 1$	$\surd$Theorem 2	$\times$Corollary 1	$\times$Theorem 4
$\mathbb{T} = \mathbb{N}$	$\surd$Theorem 2	$\surd$Theorem 8	
oERAs	$\surd$Theorem 2	$\surd$ Theorem 9	

Table 1: Summary of Section 5 ($\surd$ = decidability, $\times$ = undecidability)

4. $\forall i \in [\![0;m]\!], \forall a_i \in \Sigma \cup \Sigma_0 \implies (\exists k \in [\![0;m]\!] \exists K \subseteq [\![0;N]\!], a_k = t_K \wedge |\{j \in [\![0;i-1]\!], a_j \in \Sigma \cup \Sigma_0\}| \in K \wedge \tau_k - \tau_i = 1)$ *meaning that each of the N observations is followed by its corresponding tick exactly one time unit after it.*
5. $\forall i \in [\![0;m]\!], \forall I \subseteq [\![0;N]\!], (a_i = t_I \wedge \tau_m - \tau_i \geq 1) \implies \exists j \in [\![i+1;m]\!], \exists J \subseteq [\![0;N]\!], (I \subseteq J \wedge a_j = t_J)$ *meaning that if a clock ticked and the run is still at least one time unit long, then there will be a new tick of this clock within the rest of the run.*

We postpone the proof of this lemma to Appendix C.

Note that this lemma implies that $T([\rho])$ depends exclusively on the trace w, not on the path within the region automaton. Hence, given $[\rho']$ such that the trace of $[\rho']$ is w, we have $T([\rho']) = T([\rho])$. In particular, let B be the belief reached in $\mathcal{B}(\mathcal{A})$ with trace w. For any region $r \in B$ associated to a final location, there exists a run ρ' such that $Tr(\rho) = Tr(\rho')$ and $[\rho']$ ends in r. As $Tr(\rho) \notin \overline{Tr^{priv}}(Unfold_N(\mathcal{A}_{mem}))$ by assumption, we have that r is a region associated to a private location. Hence $B_{priv} \neq \emptyset$ and $B_{pub} = \emptyset$, thus B is a weakly discriminating belief.

Proof (Proof of Theorem 10). From Proposition 2, deciding weak and full opacity of $\mathcal{A}$ amounts to checking the existence of a discriminating belief in $\mathcal{B}(\mathcal{A})$. This is simply achieved by a reachability test in the (doubly exponential) finite automaton $\mathcal{B}(\mathcal{A})$.

7 Conclusion and perspectives

In this paper, we addressed three definitions of opacity on subclasses of TAs, to circumvent the undecidability from [14]. While undecidability remains for one-action TAs, we retrieve decidability for one-clock TAs, or over discrete time, or for observable ERAs. Our result for one-clock TAs is tight, since we showed that increasing the number of clocks leads to undecidability. We also gain decidability when the attacker can only perform a finite set of observations. We summarize the results from Section 5 in Table 1.

Perspectives include begin able to build a controller to ensure a TA is opaque, as well as investigating parametric versions of these problems, where timing constants considered as parameters (à la [3]) can be tuned to ensure opacity.

References

[1] Alur, R., Dill, D.L.: A theory of timed automata. Theoretical Computer Science **126**(2), 183–235 (Apr 1994). https://doi.org/10.1016/0304-3975(94)90010-8

[2] Alur, R., Fix, L., Henzinger, T.A.: Event-clock automata: A determinizable class of timed automata. Theoretical Computer Science **211**(1-2), 253–273 (1999). https://doi.org/10.1016/S0304-3975(97)00173-4

[3] Alur, R., Henzinger, T.A., Vardi, M.Y.: Parametric real-time reasoning. In: Kosaraju, S.R., Johnson, D.S., Aggarwal, A. (eds.) STOC. pp. 592–601. ACM, New York, NY, USA (1993). https://doi.org/10.1145/167088.167242

[4] Ammar, I., El Touati, Y., Yeddes, M., Mullins, J.: Bounded opacity for timed systems. Journal of Information Security and Applications **61**, 1–13 (Sep 2021). https://doi.org/10.1016/j.jisa.2021.102926

[5] André, É., Kryukov, A.: Parametric non-interference in timed automata. In: Li, Y., Liew, A. (eds.) ICECCS. pp. 37–42 (2020). https://doi.org/10.1109/ICECCS51672.2020.00012

[6] André, É., Lefaucheux, E., Lime, D., Marinho, D., Sun, J.: Configuring timing parameters to ensure execution-time opacity in timed automata. In: ter Beek, M.H., Dubslaff, C. (eds.) TiCSA. Electronic Proceedings in Theoretical Computer Science, vol. 392, pp. 1–26 (2023). https://doi.org/10.4204/EPTCS.392.1, invited paper.

[7] André, É., Lefaucheux, E., Marinho, D.: Expiring opacity problems in parametric timed automata. In: Ait-Ameur, Y., Khendek, F. (eds.) ICECCS. pp. 89–98 (2023). https://doi.org/10.1109/ICECCS59891.2023.00020

[8] André, É., Lime, D., Marinho, D., Sun, J.: Guaranteeing timed opacity using parametric timed model checking. ACM Transactions on Software Engineering and Methodology **31**(4), 1–36 (Oct 2022). https://doi.org/10.1145/3502851

[9] Arcile, J., André, É.: Timed automata as a formalism for expressing security: A survey on theory and practice. ACM Computing Surveys **55**(6), 1–36 (Jul 2023). https://doi.org/10.1145/3534967

[10] Barbuti, R., Francesco, N.D., Santone, A., Tesei, L.: A notion of non-interference for timed automata. Fundamenta Informaticae **51**(1-2), 1–11 (2002)

[11] Barbuti, R., Tesei, L.: A decidable notion of timed non-interference. Fundamenta Informaticae **54**(2-3), 137–150 (2003)

[12] Benattar, G., Cassez, F., Lime, D., Roux, O.H.: Control and synthesis of non-interferent timed systems. International Journal of Control **88**(2), 217–236 (2015). https://doi.org/10.1080/00207179.2014.944356

[13] Bryans, J.W., Koutny, M., Mazaré, L., Ryan, P.Y.A.: Opacity generalised to transition systems. International Journal of Information Security **7**(6), 421–435 (2008). https://doi.org/10.1007/s10207-008-0058-x

[14] Cassez, F.: The dark side of timed opacity. In: Park, J.H., Chen, H., Atiquzzaman, M., Lee, C., Kim, T., Yeo, S. (eds.) ISA. Lecture Notes in Computer Science, vol. 5576, pp. 21–30. Springer (2009). https://doi.org/10.1007/978-3-642-02617-1_3

[15] Dima, C.: Real-time automata. Journal of Automata, Languages and Combinatorics **6**(1), 3–23 (2001). https://doi.org/10.25596/jalc-2001-003

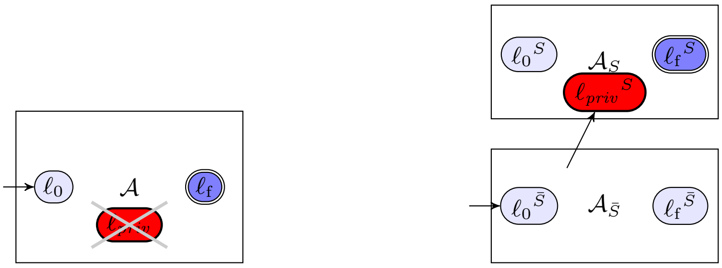
[16] Gardey, G., Mullins, J., Roux, O.H.: Non-interference control synthesis for security timed automata. Electronic Notes in Theoretical Computer Science **180**(1), 35–53 (2007). https://doi.org/10.1016/j.entcs.2005.05.046

[17] Mazaré, L.: Using unification for opacity properties. In: Ryan, P. (ed.) WITS. pp. 165–176 (Apr 2004)

[18] Ouaknine, J., Worrell, J.: On the language inclusion problem for timed automata: Closing a decidability gap. Proceedings - Symposium on Logic in Computer Science **19** (05 2004). https://doi.org/10.1109/LICS.2004.1319600

[19] Wang, L., Zhan, N.: Decidability of the initial-state opacity of real-time automata. In: Jones, C.B., Wang, J., Zhan, N. (eds.) Symposium on Real-Time and Hybrid Systems - Essays Dedicated to Professor Chaochen Zhou on the Occasion of His 80th Birthday, Lecture Notes in Computer Science, vol. 11180, pp. 44–60. Springer (2018). https://doi.org/10.1007/978-3-030-01461-2_3

[20] Wang, L., Zhan, N., An, J.: The opacity of real-time automata. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **37**(11), 2845–2856 (2018). https://doi.org/10.1109/TCAD.2018.2857363



(a) $\mathcal{A}_{pub}$

(b) $\mathcal{A}_{priv}$

Fig. 6: Illustrating $\mathcal{A}_{pub}$ and $\mathcal{A}_{priv}$

A Formal definitions

Definition 12 (Public runs automaton $\mathcal{A}_{pub}$).
Let $\mathcal{A} = (\Sigma, L, \ell_0, L_{priv}, L_f, \mathbb{X}, I, E)$ be a TA. We define the public runs TA $\mathcal{A}_{pub} = (\Sigma, L \setminus L_{priv}, \emptyset, L_f \setminus L_{priv}, \mathbb{X}, I', E')$ with I' and E' precised as follows:

1. *I' is the restriction $I|_{L \setminus L_{priv}}$ of I to the set of locations of $\mathcal{A}_{pub}$;*
2. *$E' = E \setminus \{(\ell, g, a, R, \ell') \in E \mid \ell \in L_{priv} \vee \ell' \in L_{priv}\}$ is the remaining set of transitions when private locations are removed from L.*

Example 4. We illustrate the constructions of $\mathcal{A}_{pub}$ and $\mathcal{A}_{priv}$ in Figs. 6a and 6b.

B Opacity of TAs over discrete time

Lemma 1. *The language of a discrete-time TA and the language of its region automaton are in bijection.*

Proof. Let $\mathcal{A}$ be a discrete-time TA. We explicit the bijection of the lemma.

Given a path ρ of $\mathcal{A}$ generating the timed word w, as $\mathcal{A}$ includes ticks, w is of the form

$$(t,1)\dots(t,\tau_0)(a_0,\tau_0)\;(t,\tau_0{+}1)\dots(t,\tau_1)(a_1,\tau_1)\;\dots\;(t,\tau_{n-1}{+}1)\dots(t,\tau_n)(a_n,\tau_n).$$

To the timed word w, we associate the untimed word

$$\underbrace{tt\dots t}_{t_0 \text{ times}}\; a_0 \;\underbrace{tt\dots t}_{(t_1-t_0) \text{ times}}\; a_1 \;\dots\; \underbrace{tt\dots t}_{(t_n-t_{n-1}) \text{ times}}\; a_n.$$

This untimed word is produced within the region automaton by the path corresponding to ρ. This association is injective as the sequence $(\tau_i)_{i \leq n}$ which was removed in the transformation depends only on the number of t of the timed word. Moreover, it is surjective as given an untimed word in $\mathcal{RA}_{\mathcal{A}}$

$w' = \underbrace{tt\dots t}_{k_0 \text{ times}}\ a_0\ \underbrace{tt\dots t}_{k_1 \text{ times}}\ a_1\ \dots \underbrace{tt\dots t}_{k_n \text{ times}}\ a_n$ produced by a path ρ' of the region automaton, defining

$$w = (a_0, k_0)(a_1, k_0 + k_1)\dots(a_n, \sum_{i=0}^{n} k_i)$$

we have that w is the timed word generated by the unique path of the TA corresponding to ρ' and w is associated to w'.

Proposition 1 (Decidability of language inclusion in discrete-time TAs). *Language inclusion in discrete-time TAs is decidable.*

Proof. Let $\mathcal{A}$ and $\mathcal{B}$ be two discrete-time TAs, and let $\mathcal{RA}_\mathcal{A}$ and $\mathcal{RA}_\mathcal{B}$ be their respective region automata. Then from Lemma 1, we have

$$Tr(\mathcal{A}) \subseteq Tr(\mathcal{B}) \text{ if and only if } Tr(\mathcal{RA}_\mathcal{A}) \subseteq Tr(\mathcal{RA}_\mathcal{B})$$

Thus deciding the language inclusion in discrete-time TAs amounts to solving the language inclusion problem in the context of finite regular automata, known to be decidable.

Theorem 8 (Decidability of weak and full opacity in discrete-time TAs). *Weak and full opacity are decidable for discrete-time TAs.*

Proof. Let $\mathcal{A}$ be a discrete-time automaton with private locations set L_{priv}. The construction in Section 4.1 is still compatible with discrete time clocks so we can build two discrete-time TAs $\mathcal{A}_{priv}$ and $\mathcal{A}_{pub}$ such that $Tr(\mathcal{A}_{priv}) = Tr^{priv}(\mathcal{A})$ and $Tr(\mathcal{A}_{pub}) = Tr^{\overline{priv}}(\mathcal{A})$. Then testing the weak opacity property on $\mathcal{A}$ is equivalent to testing the inclusion $Tr(\mathcal{A}_{priv}) \subseteq Tr(\mathcal{A}_{pub})$ which is decidable by Proposition 1. Therefore the weak opacity problem in discrete-time TAs is decidable.

As before, thanks to Theorem 1, we can extend this result to the full opacity problem.

C Opacity with N observations

Lemma 2. *Denoting $w = a_1, \dots a_m$ the trace of $[\rho]$, $T([\rho])$ contains exactly the words $(a_1, \tau_1)\dots(a_m, \tau_m)$ satisfying the following constraints:*

1. $\forall i \in [\![1; m]\!], (a_i \in \Sigma \cup \Sigma_t \implies \tau_i - \tau_{i-1} > 0) \wedge (a_i \in \Sigma_0 \implies \tau_i - \tau_{i-1} = 0)$ *(where $\tau_0 = 0$), meaning that only the actions of Σ_0 can be made without any delay.*
2. $\forall i, j \in [\![0; m]\!], \forall k \in [\![i + 1; j - 1]\!], \forall J, K \subseteq [\![0; N]\!], \forall I \subseteq J, (i < j \wedge a_i = t_I \wedge a_j = t_J \wedge (a_k = t_K \implies K \cap J = \emptyset)) \implies \tau_j - \tau_i = 1$*, meaning that two successive ticks of the same clocks are separated by exactly 1 time unit.*

3. $\exists i \in [\![1;m]\!], \exists I \subseteq [\![0;N]\!] \forall j \leq i, \forall J \subseteq [\![0;N]\!], a_i = t_I \wedge 0 \in I \wedge \tau_i = 1 \wedge (a_j = T_J \implies 0 \notin J)$, *meaning that the first occurrence of the tick of the clock* x_0 *is at time 1.*
4. $\forall i \in [\![0;m]\!], \forall a_i \in \Sigma \cup \Sigma_0 \implies (\exists k \in [\![0;m]\!] \exists K \subseteq [\![0;N]\!], a_k = t_K \wedge |\{j \in [\![0;i-1]\!], a_j \in \Sigma \cup \Sigma_0\}| \in K \wedge \tau_k - \tau_i = 1)$ *meaning that each of the* N *observations is followed by its corresponding tick exactly one time unit after it.*
5. $\forall i \in [\![0;m]\!], \forall I \subseteq [\![0;N]\!], (a_i = t_I \wedge \tau_m - \tau_i \geq 1) \implies \exists j \in [\![i+1;m]\!], \exists J \subseteq [\![0;N]\!], (I \subseteq J \wedge a_j = t_J)$ *meaning that if a clock ticked and the run is still at least one time unit long, then there will be a new tick of this clock within the rest of the run.*

Proof. First remark that, given a run $\rho' \in [\rho]$, the trace of ρ' must satisfy those conditions. Indeed, the first condition is ensured by the fact that in $\mathit{Unfold}_N(\mathcal{A})$ transitions labeled with an action of Σ' reset at least one clock and that, due to guards only actions from ε or Σ_0 can be taken with a clock at 0. The third condition is ensured by the fact that x_0 starts at 0 and whenever x_0 reaches 1, it is reset and a tick occurs. The remaining conditions are ensured by the fact that a new observable action will lead to the next copy of the initial TA and reset the associated clock, and the only transition that can be taken when an activated tick clock (a tick clock that has been reset by an observation) is equal to 1 is the tick transition.

Now let $\sigma = (a_1, \tau_1) \dots (a_m, \tau_m) \in T([\rho])$. The existence of a run whose trace is σ comes from two facts: (1) a classical result on the region automaton guarantees the existence of runs ending with any valuation satisfying the region's constraints and (2) the ticks make sure that the untimed sequence retain the order between the fractional parts of the timings of the observable actions.

More formally, once a prefix $\sigma_0 = (a_1, \tau_1) \dots (a_k, \tau_k)$ of σ has been observed, the value of the clocks of ticks are known: for all x_i after observing σ_0 and waiting τ time units, the valuation of x_i, denoted $v_i(\sigma_0, \tau)$, is τ plus the difference between τ_k and the last τ_j such that either $i \in a_j$ or a_j is the ith observation. We denote $[\mu]_{\sigma_0}$ the set of valuations $\mu' \in [\mu]$ such that there exists $\tau \leq \tau_{k+1} - \tau_k$ (no constraint on τ if $k = m$) such that for all $i \leq N$, $\mu'(x_i) = v_i(\sigma_0, \tau)$. We say that a set S of run satisfy every valuation of $[\mu]_{\sigma_0}$ if for every valuation $\mu' \in [\mu]_{\sigma_0}$, there exists $\rho' \in S$ whose valuation in the last configuration is equal to μ' for every clock below the maximum threshold, and is beyond the maximum threshold if μ' is.

More precisely, let $n \in \mathbb{N}$ and $[\rho_0]$ be a prefix of $[\rho]$ of length n, let $\sigma_0 = (a_1, \tau_1) \dots (a_k, \tau_k)$ be the trace of $[\rho_0]$. We show by recurrence on n that, denoting $r = (\ell, [\mu])$ the region in which $[\rho_0]$ ends then there exists a set S of paths in $[\rho_0]$ whose trace is σ_0 and satisfying every valuations of $[\mu]_{\sigma_0}$.

If $n = 0$, this is trivially true as r is the initial region.

Assume the property is true for some $n \in \mathbb{N}$, denote $[\rho_0] = [\rho_{-1}], (\ell, [\mu]), b, (\ell', [\mu'])$ and σ_{-1} be the trace of $[\rho_{-1}], (\ell, [\mu])$. By hypothesis, there is a set S of paths satisfying every valuation of $[\mu]_{\sigma_{-1}}$ and whose prefix is σ_{-1}.

If $b = \varepsilon$, extending the paths of the set S to a set S' of paths satisfying every valuation of $[\mu']_{\sigma_{-1}}$ and whose prefix is σ_{-1} can be done using the usual method described in [1] (either ε represents a transition, and the valuations do not change, or it is used to let time pass, and in this case we extend every path by waiting until every possible configuration of the next region).

If $b = a_k \in \Sigma_0$, by the first constraint of the lemma, we know that no time elapsed since the last reset of a clock tick. In particular, this means that extending the paths of S by directly taking the transition is possible (as it is possible in the region automaton) and the valuations reached by this extension are the valuations of $[\mu]_{\sigma_{-1}}$ with at least one clock reset by the transition. Due to the reset, no time can elapse in the region $(\ell', [\mu'])$, hence these extended paths satisfy every valuation of $[\mu']_{\sigma_0}$.

If $b = a_k \in \Sigma \cup \Sigma_t$, let S' be the subset of S of runs which can wait until τ_k without changing region. S' is not empty as, while the boundaries of the valuations of $[\mu]_{\sigma_{-1}}$ depend on the ticks clock, due to conditions 2 to 5 of the lemma, activated ticks occur every time unit, and thus $\tau_k - \tau_{k-1}$ (with $\tau_0 = 0$) is smaller than the maximum delay required to reach the next region boundary. We extend the paths of S' by waiting until τ_k, and taking the transition. Again, this transition produces at least one reset, so no time can elapse after it without changing region again. This extended set of paths satisfies every valuation of $[\mu']_{\sigma_0}$.

This concludes the recurrence, and in particular when $n = m$, the set we built contains at least one run whose trace is σ.