The Opacity of Timed Automata

Jie An[1,3](✉), Qiang Gao[1], Lingtai Wang[1], Naijun Zhan[1,2](✉), and Ichiro Hasuo[3]

[1] Institute of Software, Chinese Academy of Sciences, Beijing, China
{gaoqiang,wanglt,znj}@ios.ac.cn, anjie@iscas.ac.cn
[2] School of Computer Science, Peking University, Beijing, China
[3] National Institute of Informatics, Tokyo, Japan
i.hasuo@acm.org

Abstract. Opacity serves as a critical security and confidentiality property, which concerns whether an intruder can unveil a system's secret based on structural knowledge and observed behaviors. Opacity in timed systems presents greater complexity compared to untimed systems, and it has been established that opacity for timed automata is undecidable. However, the original proof cannot be applied to decide the opacity of one-clock timed automata directly. In this paper, we explore three types of opacity within timed automata: language-based timed opacity, initial-location timed opacity, and current-location timed opacity. We begin by formalizing these concepts and establishing transformation relations among them. Subsequently, we demonstrate the undecidability of the opacity problem for one-clock timed automata. Furthermore, we offer a constructive proof for the conjecture regarding the decidability of opacity for timed automata in discrete-time semantics. Additionally, we present a sufficient condition and a necessary condition for the decidability of opacity in specific subclasses of timed automata.

Keywords: Opacity · Timed opacity · Timed automata

1 Introduction

Opacity is a critical security and confidentiality property concerning information flow within systems, often utilized to describe security and privacy concerns across various scenarios. In general, it aims at safeguarding the secret information within a system from an intruder who has knowledge of the system structure but only partial observability of its behaviours.

Considering a Labelled Transition System (LTS), the secret information within it can be a set of system traces or states. An intruder observes the system behaviours, and based on the partial observations of system behaviours, the intruder estimates whether the actual behaviours contain secret information. The system is deemed opaque if for every secret run, there exists a non-secret

A. Platzer et al. (Eds.): FM 2024, LNCS 14933, pp. 620–637, 2025.
https://doi.org/10.1007/978-3-031-71162-6_32

run exhibiting identical observations. Specifically, opacity is commonly categorized into two types based on the nature of the secret information: language-based opacity and state-based opacity. A system is called *language-opaque* if an intruder with partial observability can never determine whether a trace of the system is secret based on the observations. A system is termed *initial-state opaque* if an intruder is unable to determine whether a trace starts from a secret state, and it is termed *current-state opaque* if an intruder is unable to determine whether the current trace reaches a secret state. Extensive research has been conducted on untimed systems, such as Discrete Event Systems (DES) modeled by finite-state automata. The opacity problem of finite-state automata has been proved decidable in PSPACE [24,25]. We refer to [18] for a comprehensive survey.

However, timed systems introduce a level of complexity beyond untimed systems, as they encompass not only untimed event sequences but also the timestamps associated with actions or events. Moreover, it is recognized that time poses a potential security vulnerability for systems [10,14,19]. Therefore, considering that unobservable events also take a span of time, the opacity problem of timed systems becomes intriguing and considerably more intricate.

A simple example depicted in Fig. 1 illustrates an opacity problem inherent in timed systems. In this scenario, Alice, Bob, and Carlos can exchange messages, each with varying time durations between pairs. For instance, the transmission time between Alice and Bob, as well as vice versa, ranges from 1 to 4 time units, whereas between Alice and Carlos, it spans 1 to 2 time units. Let us consider Carlos as a secret participant within the system. Meanwhile, an intruder named Eve, possessing only partial observability, can solely monitor the behaviors of Alice and Bob. For instance, consider a situation that the current real message passing is Alice $\xrightarrow{1.2}$ Carlos $\xrightarrow{2.1}$ Bob. With partial observability, what Eve observed is Alice $\xrightarrow{3.3}$ Bob. The opacity problem thus questions whether Eve can deduce Carlos's involvement in the message passing process, thereby exposing the secret behaviors. If Eve remains unaware of Carlos's participation, we conclude that the timed system is opaque to the intruder regarding the secret role of "Carlos" and the clandestine activities. This timed system is deemed non-opaque because Eve can ascertain the presence of a third participant when Eve observes that the time taken to pass messages between Alice and Bob exceeds 4 units. In essence, this scenario can be considered a special case of language-based opacity of timed systems if we view the dashed secret behaviors as a secret timed language.

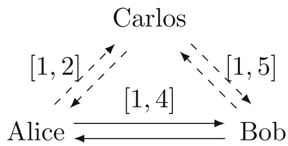

Fig. 1. A simple example for the opacity problem of timed systems

Timed automata (TA) [2], which extend finite-state automata with clock variables, are widely used as a formal model for timed systems. In a seminal work by F. Cassez [11], it was proved that the opacity problem is undecidable for TA and even for deterministic timed automata (DTA). In the proof of the undecidability for L-opacity[1] of nondeterministic timed automata (NTA), Cassez reduced the universality problem of NTA to a specific instance of the L-opacity problem of NTA. Since the universality problem for NTA is known to be undecidable [2], it logically follows that the opacity problem for NTA is also undecidable. However, in the case of one-clock timed automata (OTA), where only a single clock is involved, the universality problem becomes decidable [1]. Consequently, the reduction does not yield a conclusion on the opacity of OTA. Additionally, at the end of [11], a conjecture is given that the opacity problem of TA is decidable in the discrete-time semantics. Therefore, all these factors serve as strong motivations for us to revisit the opacity problem of timed automata.

In this paper, we investigate three types of the opacity of timed automata, i.e., *language-based timed opacity* (`LBTO`), *initial-location timed opacity* (`ILTO`), and *current-location timed opacity* (`CLTO`). These concepts are adaptations of language-based opacity, initial-state opacity, and current-state opacity to the realm of timed automata, respectively. Our main contributions are as follows.

- We formalize and compare the three types of timed opacity, and present the transformations among them, i.e., `ILTO` and `CLTO` can be reduced to `LBTO` for TA while the inverse reductions are restricted to DTA. (Sect. 3)
- We provide proof of the undecidability of the opacity problem of OTA. Following the idea in [11], it is achieved by reducing the universality problem of *OTA with epsilon transitions* to an instance of `CLTO` problem of OTA. (Sect. 4.1)
- We confirm the conjecture regarding the decidability of opacity for TA in discrete-time semantics by transforming the opacity problem into the language inclusion problem of nondeterministic finite-state automata with epsilon transitions. (Sect.4.2)
- We present both a sufficient condition and a necessary condition for the decidability of the opacity problem of specific subclasses of TA. Given a subclass of TA, a sufficient condition requires that the subclass is closed under product, complementation, and projection, and a necessary condition is that the universality problem of the subclass is decidable. (Sect. 4.3)

Related Work. Opacity problems have been extensively studied in Discrete Event Systems community [7,13,16,20,23,23,25,28,29]. We name just a few related works here. A comprehensive introduction to verification and enforcement of opacity can be found in [18]. Contrary to finite-state automata, which enjoy decidability in opacity, it has been proven that the opacity problem is undecidable for TA [11]. Therefore, various types of opacity for subclasses of TA with different restrictions have been investigated. The opacity problem of a subclass named Event-Recording Automata (ERA) [3] has also been proved undecidable in [11]. Later in [26,27], the language-based and state-based opacity problems have been proved decidable for RTA. A more comprehensive study on state-based opacity of RTA is given in [31], showing that the decision complexity is 2-EXPTIME. A kind of bounded-timed opacity is studied in [4]. Recently,

[1] It is equivalent to the current-location timed opacity (`CLTO`) defined in Sect. 3.

in [5,6], André et al. define a kind of timed opacity only considering the duration time of the executions but not the events, which is different from the classic concepts in [11]. There are also some works on the approximate opacity of Cyber-Physical Systems [21,30].

2 Preliminaries

In this section, we review the concepts of timed automata and recall several sub-classes. Let $\mathbb{N}$, $\mathbb{R}$ and $\mathbb{R}_{\geq 0}$ denote the set of natural, real and non-negative real numbers, respectively. The set of Boolean values is denoted as $\mathbb{B} = \{\top, \bot\}$, where $\top$ stands for *true* and $\bot$ for *false*. Let Σ, named alphabet, be a finite set of *events* or *actions*. Let ϵ be the special *empty action* and let $\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$.

In what follows, suppose a symbol $\mathbb{A}$ represents a class of automata, we write ϵ-$\mathbb{A}$ for the *automata with epsilon transitions*. For instance, we write ϵ-TA for TA with epsilon transitions. Also, epsilon transitions are denoted as ϵ-transitions.

2.1 Timed Words, Timed Languages and Timed Automata

A *timed word* is a finite sequence of timed actions $\omega = (\sigma_1, t_1)(\sigma_2, t_2)\cdots(\sigma_n, t_n) \in (\Sigma \times \mathbb{R}_{\geq 0})^*$, where $0 \leq t_1 \leq t_2 \leq \cdots \leq t_n$ are global timestamps, and *timed action* (σ_i, t_i) represents action σ_i occurs at time t_i for $1 \leq i \leq n$. The length of the timed word $|\omega| = n$ and the length of ϵ is 0. Particularly, a *timed word with empty action* ϵ is a sequence of timed actions and the empty action ϵ over $\Sigma_\epsilon \times \mathbb{R}_{\geq 0}$. A *timed language* $\mathcal{L}$ is a set of timed words, i.e., $\mathcal{L} \subseteq (\Sigma \times \mathbb{R}_{\geq 0})^*$.

Definition 1 (Projection). Given a subset $\Sigma_o \subseteq \Sigma$, a *projection* P_{Σ_o} on timed words w.r.t Σ_o is a function $(\Sigma \times \mathbb{R}_{\geq 0})^* \to (\Sigma_o \times \mathbb{R}_{\geq 0})^*$ s.t.

$$P_{\Sigma_o}(\epsilon) = \epsilon$$

$$P_{\Sigma_o}((\sigma, t) \cdot \omega) = \begin{cases} (\sigma, t) \cdot P_{\Sigma_o}(\omega) & \text{if } \sigma \in \Sigma_o \\ P_{\Sigma_o}(w) & \text{otherwise.} \end{cases}$$

Additionally, we extend P_{Σ_o} to timed languages, i.e., given a timed language $\mathcal{L}$, we have $P_{\Sigma_o}(\mathcal{L}) = \{P_{\Sigma_o}(\omega) \mid \omega \in \mathcal{L}\}$.

Example 1. Given a timed word $\omega = (\sigma_1, 2)(\sigma_2, 3.2)(\sigma_1, 5.7)(\sigma_3, 7)$, we have $P_{\{\sigma_1\}}(\omega) = (\sigma_1, 2)(\sigma_1, 5.7)$ and $P_{\{\sigma_2,\sigma_3\}}(\omega) = (\sigma_2, 3.2)(\sigma_3, 7)$. Note that, for timed words with empty action ϵ, say $\omega' = (\sigma_1, 2)(\epsilon, 3.2)(\sigma_1, 5.7)$, we also have $P_{\{\sigma_1\}}(\omega') = (\sigma_1, 2)(\sigma_1, 5.7)$. $\triangleleft$

Timed automata (TA) [2] extend finite-state automata with a finite set of clock variables. In each state, all clocks increase at the same rate, and a set of clocks can be reset to zero at each transition.

Let $\mathcal{C}$ be the set of clock variables and let $\Phi(\mathcal{C})$ denote the set of *clock constraints* of the form $\phi ::= \top \mid c \bowtie m \mid \phi \wedge \phi$, where $m \in \mathbb{N}$ and $\bowtie \in \{=, <, >, \leq, \geq\}$. A *clock valuation* $v : \mathcal{C} \to \mathbb{R}_{\geq 0}$ is a function assigning a non-negative real value to each clock $c \in \mathcal{C}$. $v \in \phi$ represents that the clock valuation v *satisfies* the clock constraint ϕ, i.e. ϕ evaluates to true on v. For $d \in \mathbb{R}_{\geq 0}$, let $v + d$ be the clock valuation which maps every clock $c \in \mathcal{C}$ to the value $v(c) + d$, and for a set $\mathcal{R} \subseteq \mathcal{C}$, let $[\mathcal{R} \rightarrow 0]v$ be the clock valuation which resets all clock variables in $\mathcal{R}$ to 0 and agrees with v for every clock in $\mathcal{C} \backslash \mathcal{R}$.

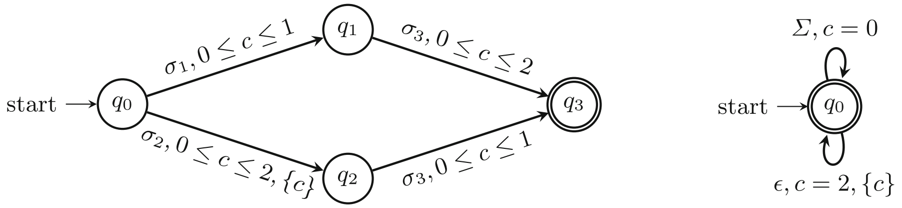

Fig. 2. An illustration for TA $\mathcal{A}$ (left side) and ϵ-TA $\mathcal{A}_\epsilon$ (right side).

Definition 2 (Timed automata). A (nondeterministic) *timed automaton* (NTA) is a 6-tuple $\mathcal{A} = (\Sigma, Q, Q_0, Q_f, \mathcal{C}, \Delta)$, where Σ is the alphabet; Q is a finite set of locations; Q_0 is a set of initial locations; Q_f is a set of accepting locations; $\mathcal{C}$ is a finite set of clocks; and $\Delta \subseteq Q \times \Sigma \times \Phi(\mathcal{C}) \times 2^{\mathcal{C}} \times Q$ is a transition relation.

A transition $(q, \sigma, \phi, \mathcal{R}, q') \in \Delta$ allows a jump from location q to q' if σ occurs and the constraint ϕ is satisfied by the current clock valuation. After that, the clocks in $\mathcal{R}$ are reset to zero, while other clocks remain unchanged.

A *state* of $\mathcal{A}$ is a pair (q, v), where $q \in Q$ is a location and v is a clock valuation. A *run* ρ of $\mathcal{A}$ over a timed word $\omega = (\sigma_1, t_1)(\sigma_2, t_2) \cdots (\sigma_n, t_n)$ is a sequence $\rho = (q_0, v_0) \xrightarrow{\tau_1, \sigma_1} (q_1, v_1) \xrightarrow{\tau_2, \sigma_2} \cdots \xrightarrow{\tau_n, \sigma_n} (q_n, v_n)$, satisfying (1) q_0 is an initial location and $v_0(c) = 0$ for each clock $c \in \mathcal{C}$; (2) for all $1 \leq i \leq n$, there is a transition $(q_{i-1}, \sigma_i, \phi_i, \mathcal{R}_i, q_i)$ such that $(v_{i-1} + \tau_i) \in \phi_i$ and $v_i = [\mathcal{R}_i \rightarrow 0](v_{i-1} + \tau_i)$; (3) $\tau_1 = t_1$ and $\tau_i = t_i - t_{i-1}$ for $2 \leq i \leq n$. Thus, each τ_i represents the delay time between the transitions. A run ρ is an *accepting* run if $q_n \in Q_f$.

The *trace* of a run ρ is the corresponding timed word $trace(\rho) = \omega$ or the empty timed word ϵ if $\rho = (q_0, v_0)$. Let $Tr_{\mathcal{A}}(q_0)$ be the set of all traces of runs from an initial location q_0 and let $Tr_{\mathcal{A}}(Q_0)$ be the set of traces of all traces of runs from any initial locations in Q_0. Additionally, given a location q and a subset $Q' \subseteq Q$, let $Tr_{\mathcal{A}}(Q_0, q)$ be the set of all traces of all runs starting from Q_0 and ending in location q, and $Tr_{\mathcal{A}}(Q_0, Q')$ be the set of all traces of all runs starting from Q_0 and ending in any locations in Q'. A timed automaton is a *deterministic timed automaton* (DTA) if $|Q_0| = 1$ and there is at most one run for each timed word.

Given a timed automaton $\mathcal{A}$, its *generated timed language* is the set of traces of runs of $\mathcal{A}$, i.e. $\mathcal{L}(\mathcal{A}) = Tr_{\mathcal{A}}(Q_0)$. The *recognized timed language* $\mathcal{L}_f(\mathcal{A})$ is the set of traces of accepting runs, i.e. $\mathcal{L}_f(\mathcal{A}) = Tr_{\mathcal{A}}(Q_0, Q_f)$.

An ϵ-NTA $\mathcal{A}_\epsilon = (\Sigma_\epsilon, Q, Q_0, Q_f, \mathcal{C}, \Delta)$ extends an NTA with ϵ-transitions in the form of $(q, \epsilon, \phi, \mathcal{R}, q')$. It can recognize timed words with ϵ over $\Sigma_\epsilon \times \mathbb{R}_{\geq 0}$.

The special empty action ϵ is viewed as invisible by default. Note that the timed language of an ϵ-NTA $\mathcal{A}_\epsilon$ is still a set of timed words defined on $(\Sigma \times \mathbb{R}_{\geq 0})^*$ [9].

Example 2. TA $\mathcal{A}$ on the left side of Fig. 2 has the unique clock c, where the alphabet $\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$. Timed word $\omega = (\sigma_2, 2)(\sigma_3, 3)$ is accepted by $\mathcal{A}$, since there is a run $\rho = q_0 \xrightarrow{2,\sigma_2} q_2 \xrightarrow{1,\sigma_3} q_3$ ending in the accepting location q_3. The recognized timed language $\mathcal{L}_f(\mathcal{A}) = \{(\sigma_1, t_1)(\sigma_3, t_2) | 0 \leq t_1 \leq 1 \wedge 0 \leq t_2 \leq 2\} \cup \{(\sigma_2, t_1)(\sigma_3, t_2) \mid 0 \leq t_1 \leq 2 \wedge 0 \leq t_2 - t_1 \leq 1\}$.

The ϵ-TA $\mathcal{A}_\epsilon$ with one clock c in Fig. 2 comes from [9]. Its generated timed language $\mathcal{L}(\mathcal{A}_\epsilon)$ is equivalent to its recognized timed language $\mathcal{L}_f(\mathcal{A}_\epsilon)$, i.e., $\mathcal{L}(\mathcal{A}_\epsilon) = \mathcal{L}_f(\mathcal{A}_\epsilon) = \{(\sigma_1, t_1) \cdots (\sigma_n, t_n) \in (\Sigma \times \mathbb{R}_{\geq 0})^* \mid \forall i \geq 0, t_i \in 2\mathbb{N} \wedge t_i \leq t_{i+1}\}$. It is clear that $P_\Sigma(\mathcal{L}(\mathcal{A}_\epsilon)) = \mathcal{L}(\mathcal{A}_\epsilon)$ and $P_\Sigma(\mathcal{L}_f(\mathcal{A}_\epsilon)) = \mathcal{L}_f(\mathcal{A}_\epsilon)$. $\lhd$

2.2 Expressiveness and Decidability of Timed Automata

Unlike finite-state automata, TA are not closed under complementation. Moreover, the universality problem (i.e., whether $\mathcal{L}_f(\mathcal{A}) = (\Sigma \times \mathbb{R}_{\geq 0})^*$), inclusion problem (i.e., whether $\mathcal{L}_f(\mathcal{A}_1) \subseteq \mathcal{L}_f(\mathcal{A}_2)$), and equivalence problem (i.e., whether $\mathcal{L}_f(\mathcal{A}_1) = \mathcal{L}_f(\mathcal{A}_2)$) are proven undecidable for TA, nonetheless, decidable for DTA [2]. Consequently, various subclasses of TA with different restrictions have been introduced and extensively studied. In the following discussion, we will revisit some of these subclasses and provide a summary of their expressiveness.

We denote one-clock timed automata as OTA and refer to nondeterministic and deterministic OTA as NOTA and DOTA, respectively. The expressive power of NOTA strictly exceeds that of DOTA, i.e., DOTA $\subset$ NOTA. However, NOTA and DTA are *incomparable*. On one hand, there exist DTA languages that elude recognition by any NOTA. Conversely, NOTA lacks closure under complementation, while DTA retains closure. There exist NOTA languages that cannot be captured by any DTA. OTA with ϵ-transitions is denoted as ϵ-OTA.

Real-timed automata (RTA) [12] is a subclass of timed automata with a single clock resetting at every transition, resulting in RTA $\subset$ DOTA. Notably, any nondeterministic RTA can be determinized, thereby endowing deterministic RTA with the same expressive power as their nondeterministic counterparts. Additionally, RTA exhibits closure properties under product, complementation, and projection, as demonstrated in [12,27].

Event-recording automata (ERA) [3] is a kind of timed automata associating each action σ with a clock to record the time length from the last occurrence of σ to the current. As ERA is a class of *determinizable* timed automata, we have ERA $\subset$ DTA. However, ERA and RTA are *incomparable*. This distinction arises because RTA may accept languages consisting of two actions separated by an interval with integer length while ERA may not.

As shown in [2], NTA $\subset$ ϵ-NTA, since that ϵ-transitions will increase the expressive power if they reset clocks [9]. For example, in Fig. 2, the timed language of $\mathcal{A}_\epsilon$ can not be represented by any NTA.

In summary, the comparable expressive power among them is in the following order RTA $\subset$ DOTA $\subset$ DTA $\subset$ NTA $\subset$ ϵ-NTA. Note that we will ignore the character 'N' in general, such as NTA = TA and NOTA = OTA.

3 Opacity Problems of Timed Automata

In this section, we investigate three types of timed opacity, i.e., *language-based timed opacity* (`LBTO`), *initial-location timed opacity* (`ILTO`) and *current-location timed opacity* (`CLTO`), and demonstrate the transformations between them.

3.1 Language-Based and Location-Based Timed Opacity

Given a TA $\mathcal{A} = (\Sigma, Q, Q_0, Q_f, \mathcal{C}, \Delta)$, an observable alphabet $\Sigma_o \subseteq \Sigma$, and a *secret timed language* $\mathcal{L}_s$, we define `LBTO` as follows.

Definition 3 (Language-based timed opacity, `LBTO`). $\mathcal{A}$ is *language-based (strongly) timed opaque* w.r.t Σ_o and $\mathcal{L}_s$ iff

$$\forall \omega \in \mathcal{L}(\mathcal{A}) \cap \mathcal{L}_s, \exists \omega' \in \mathcal{L}(\mathcal{A}) \setminus \mathcal{L}_s \text{ s.t. } P_{\Sigma_o}(\omega) = P_{\Sigma_o}(\omega') \tag{1}$$

which is equivalent to $P_{\Sigma_o}(\mathcal{L}(\mathcal{A}) \cap \mathcal{L}_s) \subseteq P_{\Sigma_o}(\mathcal{L}(\mathcal{A}) \setminus \mathcal{L}_s)$.

`LBTO` requires that for each secret trace, there exists a non-secret trace such that their observations w.r.t the observable alphabet Σ_o are identical.

Let us consider a *secret set of locations* $Q_s \subseteq Q$ within $\mathcal{A}$, instead of a secret timed language $\mathcal{L}_s$. We define `ILTO` and `CLTO` as follows.

Definition 4 (Initial-location timed opacity, `ILTO`). $\mathcal{A}$ is *initial-location timed opaque* w.r.t Σ_o and $Q_s \subseteq Q_0$ iff

$$\forall \omega \in Tr_{\mathcal{A}}(Q_s), \exists \omega' \in Tr_{\mathcal{A}}(Q_0 \setminus Q_s) \ s.t. \ P_{\Sigma_o}(\omega) = P_{\Sigma_o}(\omega') \tag{2}$$

which is equivalent to $P_{\Sigma_o}(Tr_{\mathcal{A}}(Q_s)) \subseteq P_{\Sigma_o}(Tr_{\mathcal{A}}(Q_0 \setminus Q_s))$.

`ILTO` requires that for each trace starting from a secret location, there exists a trace starting from a non-secret location such that their observations w.r.t Σ_o are identical.

Definition 5 (Current-location timed opacity, `CLTO`). $\mathcal{A}$ is *current-location timed opaque* w.r.t Σ_o and $Q_s \subseteq Q$ iff

$$\forall \omega \in Tr_{\mathcal{A}}(Q_0, Q_s), \exists \omega' \in Tr_{\mathcal{A}}(Q_0, Q \setminus Q_s) \ s.t. \ P_{\Sigma_o}(\omega) = P_{\Sigma_o}(\omega') \tag{3}$$

which is equivalent to $P_{\Sigma_o}(Tr_{\mathcal{A}}(Q_0, Q_s)) \subseteq P_{\Sigma_o}(Tr_{\mathcal{A}}(Q_0, Q \setminus Q_s))$.

`CLTO` requires that for each trace reaching a secret location, there exists a trace reaching a non-secret location such that their observations w.r.t Σ_o are identical.

Example 3. In Fig. 2, suppose $\Sigma_o = \{\sigma_3\}$ and $\mathcal{L}_s = \{(\sigma_2, t_1)(\sigma_3, t_2) \mid 0 \le t_1 \le 2 \wedge 0 \le t_2 \le 3\}$, then $\mathcal{A}$ is not LBTO w.r.t $\mathcal{L}_s$ and Σ_o: If the intruder observes a 'σ_3' at time 3, they can infer that the previous action must have been 'σ_2' rather than 'σ_1', as there is no non-secret trace with an observation of 'σ_3' at time 3.

If we consider the opacity of the corresponding untimed system, the system language is $L = \{\sigma_1, \sigma_2, \sigma_1\sigma_3, \sigma_2\sigma_3\}$ and the secret language is $L_s = \{\sigma_2\sigma_3\}$. If the current observation is σ_3, the intruder cannot ascertain whether the actual behavior is $\sigma_1\sigma_3$ or $\sigma_2\sigma_3$. Therefore, the corresponding untimed system exhibits opacity. This illustrates that timed opacity presents a distinct and intriguingly more complex challenge compared to untimed systems. $\lhd$

3.2 Transformation Between LBTO, ILTO and CLTO

We first present the transformations from ILTO to LBTO and from CLTO to LBTO with TA. Subsequently, we elucidate the reverse transformations from LBTO to ILTO and CLTO restricting to DTA.

Drawing from a common assumption in untimed systems' opacity, where a secret language is recognized by a finite-state automaton, we suppose that $\mathcal{L}_s$ can be recognized by a secret TA $\mathcal{A}_s$, i.e. $\mathcal{L}_s = \mathcal{L}_f(\mathcal{A}_s)$. The assumption is reasonable, given that every finite set of timed words can be modelled by a TA and every regular timed language can be recognized by a TA.

From ILTO to LBTO. Given a TA $\mathcal{A} = \{\Sigma, Q, Q_0, Q_f, \mathcal{C}, \Delta\}$, and a secret subset of locations $Q_s \subseteq Q_0$, the ILTO problem w.r.t Q_s and Σ_o formalized by (2) can be transformed to an LBTO problem as follows.

We first construct a TA $\mathcal{A}_s = \{\Sigma, Q, Q_0', Q_f', \mathcal{C}, \Delta\}$. Let $Q_0' = Q_s$ and mark all locations as the accepting locations $Q_f' = Q$. Then we have $\mathcal{L}(\mathcal{A}_s) = \mathcal{L}_f(\mathcal{A}_s)$. Note that $Tr_{\mathcal{A}}(Q_s) = Tr_{\mathcal{A}_s}(Q_s)$. Let $\mathcal{L}_s = \mathcal{L}_f(\mathcal{A}_s)$ be the secret timed language. Then we have

$$\begin{aligned}\mathcal{L}(\mathcal{A}) \cap \mathcal{L}_s &= \mathcal{L}(\mathcal{A}) \cap \mathcal{L}_f(\mathcal{A}_s) = \mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{A}_s) = \mathcal{L}(\mathcal{A}_s) = Tr_{\mathcal{A}_s}(Q_s) = Tr_{\mathcal{A}}(Q_s)\\ \mathcal{L}(\mathcal{A}) \setminus \mathcal{L}_s &= \mathcal{L}(\mathcal{A}) \setminus \mathcal{L}_f(\mathcal{A}_s) = \mathcal{L}(\mathcal{A}) \setminus \mathcal{L}(\mathcal{A}_s) = Tr_{\mathcal{A}}(Q_0) \setminus Tr_{\mathcal{A}_s}(Q_s)\\ &= Tr_{\mathcal{A}}(Q_0) \setminus Tr_{\mathcal{A}}(Q_s) = Tr_{\mathcal{A}}(Q_0 \setminus Q_s)\end{aligned}$$

Hence, it is transformed to the following LBTO problem of $\mathcal{A}$ w.r.t $\mathcal{L}_s$ and Σ_o

$$\forall \omega \in \mathcal{L}(\mathcal{A}) \cap \mathcal{L}_s, \exists \omega' \in \mathcal{L}(\mathcal{A}) \setminus \mathcal{L}_s \text{ s.t. } P_{\Sigma_o}(\omega) = P_{\Sigma_o}(\omega')$$

$\square$

From CLTO to LBTO. Given a TA $\mathcal{A} = \{\Sigma, Q, Q_0, Q_f, \mathcal{C}, \Delta\}$, and $Q_s \subseteq Q$, the CLTO problem w.r.t Q_s and Σ_o formalized by (3) can be transformed to an LBTO problem as follows.

We can construct a TA $\mathcal{A}' = \{\Sigma, Q, Q_0, Q_f', \mathcal{C}, \Delta\}$ which is a copy of $\mathcal{A}$ except that the accepting locations are changed from Q_f to Q_s, i.e. $Q_f' = Q_s$.

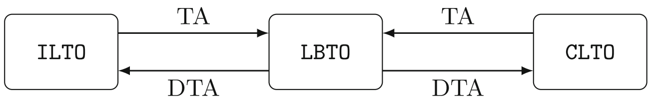

Fig. 3. The transformation between LBTO, ILTO, and CLTO.

Therefore, we have $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$, i.e., $Tr_{\mathcal{A}}(Q_0) = Tr_{\mathcal{A}'}(Q_0)$. Let $\mathcal{L}_s = \mathcal{L}_f(\mathcal{A}')$ be the secret language, then we have

$$\mathcal{L}(\mathcal{A}') \cap \mathcal{L}_s = \mathcal{L}_s = Tr_{\mathcal{A}'}(Q_0, Q'_f) = Tr_{\mathcal{A}'}(Q_0, Q_s)$$
$$\mathcal{L}(\mathcal{A}') \setminus \mathcal{L}_s = Tr_{\mathcal{A}'}(Q_0) \setminus Tr_{\mathcal{A}'}(Q_0, Q'_f) = Tr_{\mathcal{A}'}(Q_0, Q \setminus Q'_f) = Tr_{\mathcal{A}'}(Q_0, Q \setminus Q_s)$$

Hence, it is transformed to the following LBTO problem of $\mathcal{A}'$ w.r.t $\mathcal{L}_s$ and Σ_o

$$\forall \omega \in \mathcal{L}(\mathcal{A}') \cap \mathcal{L}_s, \exists \omega' \in \mathcal{L}(\mathcal{A}') \setminus \mathcal{L}_s \text{ s.t. } P_{\Sigma_o}(\omega) = P_{\Sigma_o}(\omega')$$

□

From LBTO to CLTO. Given a DTA $\mathcal{A} = \{\Sigma, Q, Q_0, Q_f, \mathcal{C}, \Delta\}$, and a secret DTA $\mathcal{A}_s$ and let $\mathcal{L}_s = \mathcal{L}_f(\mathcal{A}_s)$, the LBTO problem w.r.t $\mathcal{L}_s$ and Σ_o formalized by (1) can be transformed to a CLTO problem as follows.

We construct a timed automaton $\mathcal{A}' = (\Sigma, Q', Q'_0, Q'_f, \mathcal{C}', \Delta')$ in the following steps. We first make a copy of $\mathcal{A}$ as $\mathcal{A}'' = (\Sigma, Q, Q_0, Q''_f, \mathcal{C}, \Delta)$ and let all locations be the accepting locations $Q''_f = Q$. We have $\mathcal{L}_f(\mathcal{A}'') = \mathcal{L}(\mathcal{A})$. Since DTA are closed under product and complementation [2], we construct a product TA $\mathcal{A}_p = \mathcal{A}'' \times \mathcal{A}_s$ and then construct a product TA $\mathcal{A}'_p = \mathcal{A}'' \times \overline{\mathcal{A}_p}$. Therefore, we have

$$\mathcal{L}_f(\mathcal{A}_p) = \mathcal{L}_f(\mathcal{A}'') \cap \mathcal{L}_f(\mathcal{A}_s) = \mathcal{L}(\mathcal{A}) \cap \mathcal{L}_s$$
$$\mathcal{L}_f(\mathcal{A}'_p) = \mathcal{L}_f(\mathcal{A}'') \cap (\overline{\mathcal{L}(\mathcal{A})} \cup \overline{\mathcal{L}_s}) = \mathcal{L}(\mathcal{A}) \cap \overline{\mathcal{L}_s} = \mathcal{L}(\mathcal{A}) \setminus \mathcal{L}_s.$$

Let $\mathcal{A}' = \mathcal{A}_p \cup \mathcal{A}'_p$ and let Q_s be the set of accepting locations of $\mathcal{A}_p$. We denote by $Q_f^{\mathcal{A}'_p}$ the set of accepting locations of $\mathcal{A}'_p$. It is clear that $Q_f^{\mathcal{A}'_p} \subset Q' \setminus Q_s$. Therefore, it is transformed to the following CLTO problem w.r.t Q_s and Σ_o

$$\forall \omega \in Tr_{\mathcal{A}'}(Q'_0, Q_s), \exists \omega' \in Tr_{\mathcal{A}'}(Q'_0, Q_f^{\mathcal{A}'_p}) \; s.t. \; P_{\Sigma_o}(\omega) = P_{\Sigma_o}(\omega').$$

□

From LBTO to ILTO. The reduction is similar to the above reduction from LBTO to CLTO. Similar to [28], we suppose that $\mathcal{L}_s$ and $\mathcal{L}(\mathcal{A}) \setminus \mathcal{L}_s$ are both prefix-closed. Then we can build two DTA $\mathcal{A}_1$ and $\mathcal{A}_2$ such that $\mathcal{L}(\mathcal{A}_1) = \mathcal{L}_f(\mathcal{A}_p)$ and $\mathcal{L}(\mathcal{A}_2) = \mathcal{L}_f(\mathcal{A}'_p)$. Let $\mathcal{A}' = \mathcal{A}_1 \cup \mathcal{A}_2$ and let the secret set Q_s be the initial location set of $\mathcal{A}_1$. Then, the LBTO problem is transformed to the following ILTO problem w.r.t Q_s and Σ_o

$$\forall \omega \in Tr_{\mathcal{A}'}(Q_s), \exists \omega' \in Tr_{\mathcal{A}'}(Q'_0 \setminus Q_s) \; s.t. \; P_{\Sigma_o}(\omega) = P_{\Sigma_o}(\omega').$$

□

Figure 3 summarizes the transformation between `LBTO`, `ILTO`, and `CLTO`. Since the complementation operation is involved in the transformations from `LBTO` to `CLTO` and to `ILTO`, we argue that the two transformations do not hold for general TA. Nevertheless, it is enough for supporting the results presented in Sect. 4.

4 Decidability and Undecidability of Timed Opacity Problems

This section serves to establish key results regarding the undecidability of opacity problems for OTA, the decidability of opacity problems for TA in discrete-time semantics, and a sufficient condition and a necessary condition for the decidability of opacity problems within various subclasses of TA. Consequently, our findings bridge a gap in the decidability of timed opacity problems and provide constructive proof of the conjecture proposed in [11]. These conditions delineate the system properties essential for designing opaque timed systems.

4.1 Undecidability of Opacity Problems of OTA

We first consider the `CLTO` problem of OTA and prove its undecidability. Moreover, our proof also holds for DOTA. Therefore, based on the transformations shown in Sect. 3.2, the three types of opacity problems of DOTA, OTA, and ϵ-OTA are all proven undecidable. The detailed proofs are presented as follows.

Lemma 1. *Given a OTA $\mathcal{A} = (\Sigma, Q, Q_0, Q_f, \{c\}, \Delta)$ and an observable alphabet $\Sigma_o \subset \Sigma$, there is an ϵ-OTA $\mathcal{A}'$ s.t. $\mathcal{A}$ is* `CLTO` *iff $\mathcal{A}'$ is* `CLTO`*.*

Proof. The ϵ-OTA $\mathcal{A}' = (\Sigma' \cup \{\epsilon\}, Q, Q_0, Q_f, \{c\}, \Delta')$ can be built as follows. Build a new alphabet Σ' s.t. $\Sigma_o \subset \Sigma' \subset \Sigma$. Suppose $\Sigma \setminus \Sigma' = \{\sigma'_1, \sigma'_2, \cdots, \sigma'_n\}$, the transition set Δ' is constructed from Δ by replacing σ'_i with ϵ for each transition $(q, \sigma'_i, \phi, \mathcal{R}, q') \in \Delta$.

Since each σ'_i is an unobservable action, i.e., $\sigma'_i \notin \Sigma_o$, it is equivalent to ϵ w.r.t the timed opacity problem with projection P_{Σ_o}. After replacing the corresponding transitions with ϵ-transitions, checking `CLTO` of OTA $\mathcal{A}$ is equivalent to checking `CLTO` of ϵ-OTA $\mathcal{A}'$. □

The following lemma follows the proof idea in [11]. The difference is that we reduce the universality problem of ϵ-NTA, instead of NTA, to a `CLTO` problem.

Lemma 2. *Given an ϵ-NTA $\mathcal{A}_\epsilon = \{\Sigma \cup \{\epsilon\}, Q, Q_0, Q_f, \mathcal{C}, \Delta\}$, there is an NTA $\mathcal{A}'$ s.t. the universality problem of $\mathcal{A}_\epsilon$ is equivalent to the* `CLTO` *problem of $\mathcal{A}'$.*

Proof. Given ϵ-NTA $\mathcal{A}_\epsilon$, the universality problem asks if $\mathcal{L}_f(\mathcal{A}_\epsilon) = (\Sigma \times \mathbb{R}_{\geq 0})^*$. We first introduce a new non-accepting location $\tilde{q}$ and then build its complete ϵ-NTA $\tilde{\mathcal{A}}_\epsilon$, where the location set $\tilde{Q} = Q \cup \{\tilde{q}\}$ and the accepting locations are unchanged. We have $\mathcal{L}_f(\tilde{\mathcal{A}}_\epsilon) = \mathcal{L}_f(\mathcal{A}_\epsilon)$ and $\mathcal{L}(\tilde{\mathcal{A}}_\epsilon) = (\Sigma \times \mathbb{R}_{\geq 0})^*$. Based on $\tilde{\mathcal{A}}_\epsilon$, we build an NTA $\mathcal{A}' = (\Sigma', \tilde{Q}, Q_0, Q_f, \mathcal{C}, \Delta')$ by introducing an action $a \notin \Sigma$, i.e.,

$\Sigma' = \Sigma \cup \{a\}$ and replacing all ϵ-transitions in $\tilde{\mathcal{A}}_\epsilon$ with a-transitions. It is clear that $P_\Sigma(\mathcal{L}(\mathcal{A}')) = \mathcal{L}(\tilde{\mathcal{A}}_\epsilon) = (\Sigma \times \mathbb{R}_{\geq 0})^*$ and $P_\Sigma(\mathcal{L}_f(\mathcal{A}')) = P_\Sigma(\mathcal{L}_f(\tilde{\mathcal{A}}_\epsilon))$. Let the secret set $Q_s = \tilde{Q} \setminus Q_f$ and the observable alphabet $\Sigma_o = \Sigma$, the universality problem of $\mathcal{A}_\epsilon$ equals to the CLTO problem of $\mathcal{A}'$ w.r.t Q_s and Σ_o. □

The proof of Lemma 2 is not related to the number of clocks, so the universality problem of ϵ-OTA can be reduced to the CLTO problem of OTA. According to [1], the former problem is undecidable.

Theorem 1. *The* CLTO *problems of OTA and ϵ-OTA are undecidable.*

Note that the reduction in Lemma 1 does not depend on the nondeterministic property. Therefore, it works for DOTA, i.e., given a DOTA $\mathcal{A}$, there is an ϵ-OTA $\mathcal{A}'$ s.t. $\mathcal{A}$ is CLTO iff $\mathcal{A}'$ is CLTO. Then by Theorem 1, the CLTO of DOTA is also undecidable. Depending on the transformation in Sect. 3.2, we have the conclusion.

Theorem 2. *The* LBTO*,* ILTO*, and* CLTO *problems of DOTA, OTA, and ϵ-OTA are all undecidable.*

4.2 Decidability in the Discrete-Time Semantics

The above discussions are under the continuous-time semantics. This section provides a constructive proof confirming the conjecture in [11] that language-based timed opacity of TA is decidable under discrete-time semantics, i.e., the time domain is $\mathbb{N}$.

At first, we introduce several concepts under the discrete-time semantics. In an *integral timed word* ω over $\Sigma \times \mathbb{N}$, all events have integral timestamps. An *integral timed language* L is a set of integral timed words, i.e., $L \subseteq (\Sigma \times \mathbb{N})^*$. Given a TA $\mathcal{A}$ under discrete-time semantics, the generated and recognized timed languages, denoted by $L(\mathcal{A})$ and $L_f(\mathcal{A})$, are integral timed languages. A function $\mathit{Tick} : (\Sigma \times \mathbb{N})^* \to (\Sigma \cup \{\checkmark\})^*$ maps an integral timed word to an untimed word over $\Sigma \cup \{\checkmark\}$.

The basic proof idea is as follows. Under the discrete-time semantics, by Definition 3, the LBTO problem is equivalent to the inclusion problem between the projections of two integral timed languages. According to [22], every integral timed language corresponds to an untimed *Tick* language, therefore we first build an integral automaton $\mathcal{A}^\checkmark$ accepting the integral timed language via the *Tick* language. Then, based on $\mathcal{A}^\checkmark$, we construct a nondeterministic finite-state automaton with ϵ-transitions (ϵ-NFA) accepting the projection of the integral timed language via the *Tick* language. *Therefore, we transform the* LBTO *problem to the language inclusion problem of ϵ-NFA, which is decidable.*

Definition 6 (Tick). Given an integral timed word $\omega = (\sigma_1, t_1)(\sigma_2, t_2)\ldots(\sigma_n, t_n)$, $t_i \in \mathbb{N}$ for $1 \leq i \leq n$, $\mathit{Tick}(\omega) = \underbrace{\checkmark \ldots \checkmark}_{t_1} \sigma_1 \cdots \underbrace{\checkmark \ldots \checkmark}_{t_i - t_{i-1}} \sigma_i \cdots \sigma_n \in (\Sigma \cup \{\checkmark\})^*$.

Hence, the number of ✓ between two events in the untimed word $Tick(\omega)$ is equal to the delay time length between two events in the timed word ω. For example, let $\omega = (\sigma_1, 2)(\sigma_2, 3)$, we have $Tick(\omega) = \checkmark\checkmark\sigma_1\checkmark\sigma_2$. We also extend *Tick* to the integral timed languages, i.e., $Tick(L) = \{Tick(\omega) \mid \omega \in L\}$. We call the untimed language $Tick(L)$ as *Tick language*.

Therefore, we can transform the `LBTO` problem under discrete-time semantics into the inclusion problem of the corresponding *Tick* languages.

Lemma 3. *Given the* `LBTO` *problem w.r.t* $L(\mathcal{A})$ *and* L_s*, we have* $P_{\Sigma_o}(L(\mathcal{A}) \cap L_s) \subseteq P_{\Sigma_o}(L(\mathcal{A}) \setminus L_s) \Leftrightarrow Tick(P_{\Sigma_o}(L(\mathcal{A}) \cap L_s)) \subseteq Tick(P_{\Sigma_o}(L(\mathcal{A}) \setminus L_s))$.

In the following, we present a procedure to construct an ϵ-NFA recognizing the *Tick*-language of the projection of the integral timed language of a given timed automaton $\mathcal{A}$.

According to [22], given a TA $\mathcal{A}$, we build an *integral automaton* (IA) recognizing the integral timed language of $\mathcal{A}$. The basic idea is to discretize the real-valued clock valuations based on the concept of *region equivalence* [2,8].

Let $\kappa : \mathcal{C} \to \mathbb{N}$ be the ceiling function, i.e., $\kappa(c)$ is the maximal integer constant appearing in the clock constraints of clock c on transitions. For $d \in \mathbb{R}$, let $\lfloor d \rfloor$ denote the integer part of d, and let $frac(d)$ denote the fractional part.

Definition 7 (Region equivalence [2,8]). Two clock valuations $v_1, v_2 : \mathcal{C} \to \mathbb{R}_{\geq 0}$ are region-equivalent, denoted by $v_1 \cong v_2$ iff

1. $\forall c \in \mathcal{C}$, either $\lfloor v_1(c) \rfloor = \lfloor v_2(c) \rfloor$, or $v_1(c) > \kappa(c) \wedge v_2(c) > \kappa(c)$.
2. $\forall c \in \mathcal{C}$, if $v_1(c) \leq \kappa(c)$, then $frac(v_1(c)) = 0$ iff $frac(v_2(c)) = 0$.
3. $\forall c_1, c_2 \in \mathcal{C}$, if $v_1(c_1) \leq \kappa(c_1) \wedge v_1(c_2) \leq \kappa(c_2)$, then $frac(v_1(c_1)) \leq frac(v_1(c_2))$ iff $frac(v_2(c_1)) \leq frac(v_2(c_2))$.

A *region* $[v] = \{\forall v' : \mathcal{C} \to \mathbb{R}_{\geq 0} \mid v' \cong v\}$ is an equivalence class induced by region equivalence $\cong$, which denotes the set of all clock valuations v' region-equivalent to v. Given a TA $\mathcal{A}$, we denote by $Reg(\mathcal{A})$ the set of regions. According to [2], $Reg(\mathcal{A})$ is finite and $|Reg(\mathcal{A})|$ is bounded by $|\mathcal{C}|! \cdot 2^{|\mathcal{C}|} \cdot \prod_{c \in \mathcal{C}}(2\kappa(c) + 2)$. Specifically, we denote by $IReg(\mathcal{A})$ the set of regions only contain the integer numbers, i.e. $IReg(\mathcal{A}) = \{[v] \mid \forall c \in \mathcal{C},\ v(c) \in \{0, 1, ..., \kappa(c) + 1\}\}$. According to region equivalence, there is only one element v in a region $[v] \in IReg(\mathcal{A})$.

Definition 8 (Integral automata). Given a TA $\mathcal{A} = (\Sigma, Q, Q_0, Q_f, \mathcal{C}, \Delta)$, an integral automaton (IA) $\mathcal{A}^{\checkmark} = (\Sigma \cup \{\checkmark\}, Q^{\checkmark}, Q_0^{\checkmark}, Q_f^{\checkmark}, \Delta^{\checkmark})$ can be constructed as follows: the finite set of locations $Q^{\checkmark} = Q \times IReg(\mathcal{A})$; the set of initial locations $Q_0^{\checkmark} = Q_0 \times \{[\mathbf{0}]\}$; the set of accepting locations $Q_f^{\checkmark} = Q_f \times IReg(\mathcal{A})$; and the transition relation $\Delta^{\checkmark} \subseteq Q^{\checkmark} \times \Sigma \cup \{\checkmark\} \times Q^{\checkmark}$ includes σ-translations and ✓-translations constructed based on transitions $(q, \sigma, \phi, \mathcal{R}, q') \in \Delta$:

- σ-translation: $(q, [v]) \xrightarrow{\sigma} (q', [v'])$, s.t. $\exists [v], [v'] \in IReg(\mathcal{A}),\ v \in \phi$ *and* $v' = [\mathcal{R} \to 0]v$.
- ✓-translation: $(q, [v]) \xrightarrow{\checkmark} (q, [v'])$, s.t. $\exists [v], [v'] \in IReg(\mathcal{A}),\ v' = v + 1$.

A σ-translation represents a discrete jump from a symbolic state (location) $(q, [v])$ to a symbolic state $(q', [v'])$. It simulates the transition $(q, \sigma, \phi, \mathcal{R}, q')$ in TA $\mathcal{A}$ but only triggered by the clock valuations containing integral assignments. A $\checkmark$-translation simulates the one time-unit passing in a location of $\mathcal{A}$. The generated and recognized languages, denoted by $L(\mathcal{A}^{\checkmark})$ and $L_f(\mathcal{A}^{\checkmark})$, are untimed languages over $\Sigma \cup \{\checkmark\}$.

The following lemma states that the corresponding IA $\mathcal{A}^{\checkmark}$ recognizes the integral timed language of TA $\mathcal{A}$ via the *Tick* language.

Lemma 4 (Proposition 10 in [22]). *Given a TA $\mathcal{A}$, there exists an IA $\mathcal{A}^{\checkmark}$ whose language $L_f(\mathcal{A}^{\checkmark})$ is equivalent to $Tick(L_f(\mathcal{A}))$.*

ϵ-NFA Construction. Based on $\mathcal{A}^{\checkmark}$, we can construct an ϵ-NFA $\mathcal{A}^{\checkmark}_{\Sigma_o}$ that can accept the *Tick* language of the projection of the integral timed language of $\mathcal{A}$, i.e. $Tick(P_{\Sigma_o}(L_f(\mathcal{A})))$, by the following two steps.

1. Replace all $\sigma \notin \Sigma_o$ with ϵ.
2. For all traces that end up in $Q^{\checkmark}_f$ and contain only ϵ-translations and $\checkmark$-translations, construct a fresh set of ϵ-transitions Δ_ϵ by
   - Introducing a fresh location q_s as the unique accepting location.
   - For all $q \in Q^{\checkmark}$ s.t. $q \in Q^{\checkmark}_0$ or exist $(q', \sigma, q) \in \Delta^{\checkmark}$ with $\sigma \in \Sigma_o$, if (1) $q \in Q^{\checkmark}_f$ or (2) there exists a transition sequence from q to some location $q'' \in Q^{\checkmark}_f$ that only contains $\{\epsilon, \checkmark\}$-transitions, then adding an ϵ-transition (q, ϵ, q_s) into Δ_ϵ.

Therefore, we construct an ϵ-NFA $\mathcal{A}^{\checkmark}_{\Sigma_o} = (\Sigma^{\checkmark^{\Sigma_o}}, Q^{\checkmark^{\Sigma_o}}, Q_0^{\checkmark^{\Sigma_o}}, Q_f^{\checkmark^{\Sigma_o}}, \Delta^{\checkmark^{\Sigma_o}})$, where the alphabet $\Sigma^{\checkmark^{\Sigma_o}} = \Sigma_o \cup \{\epsilon, \checkmark\}$; the set of locations $Q^{\checkmark^{\Sigma_o}} = Q^{\checkmark} \cup \{q_s\}$; the set of initial locations $Q_0^{\checkmark^{\Sigma_o}} = Q^{\checkmark}_0$; the set of accepting locations $Q_f^{\checkmark^{\Sigma_o}} = \{q_s\}$; and the set of transitions $\Delta^{\checkmark^{\Sigma_o}} = \{(q, \sigma, q') \in \Delta^{\checkmark} \mid \sigma \in \Sigma_o \cup \{\checkmark\}\} \cup \{(q, \epsilon, q') \mid (q, \sigma, q) \in \Delta^{\checkmark} \wedge \sigma \notin \Sigma_o\} \cup \Delta_\epsilon$.

Lemma 5. *Given a TA $\mathcal{A}$, the language of the constructed ϵ-NFA $\mathcal{A}^{\checkmark}_{\Sigma_o}$ is equivalent to the Tick language of the projection of the integral timed language of $\mathcal{A}$, i.e., $L_f(\mathcal{A}^{\checkmark}_{\Sigma_o}) = Tick(P_{\Sigma_o}(L_f(\mathcal{A})))$.*

Given a TA $\mathcal{A}$ and a secret TA $\mathcal{A}_s$ under the discrete-time semantics, let $L_s = L_f(\mathcal{A}_s)$, by Lemma 4 and Lemma 5, we can always build two ϵ-NFA A_1 and A_2 such that $L_f(A_1) = Tick(P_{\Sigma_o}(L(\mathcal{A}) \cap L_s))$ and $L_f(A_2) = Tick(P_{\Sigma_o}(L(\mathcal{A}) \setminus L_s))$, since TA in the discrete-time semantics are closed under product and complementation [15]. Hence, by Lemma 3, the `LBTO` problem w.r.t the integral timed languages $L(\mathcal{A})$ and L_s can be transformed into the language inclusion problem between ϵ-NFA A_1 and A_2, and the latter is decidable in PSPACE-complete [17]. Therefore, we have the following conclusion.

Theorem 3. *The* `LBTO`*,* `ILTO`*, and* `CLTO` *of TA under the discrete-time semantics are decidable.*

4.3 Sufficient Condition and Necessary Condition

Given a subclass of TA, denoted by $\mathcal{X}$-automata, we present a sufficient condition and a necessary condition on the decidability of opacity problems of $\mathcal{X}$-automata. According to the transformation in Fig. 3, LBTO is the strongest property, i.e., ILTO and CLTO can be reduced to LBTO. Hence, we consider the sufficient condition of LBTO. For the necessary condition, we consider the CLTO problem.

Sufficient Condition of LBTO. Given an $\mathcal{X}$-automaton X, and a secret language $\mathcal{L}_s$ which can be recognized by a secret $\mathcal{X}$-automaton X_s, i.e., $\mathcal{L}_s = \mathcal{L}_f(X_s)$, by Definition 3, the LBTO problem asks if $\forall \omega \in \mathcal{L}(X) \cap \mathcal{L}_f(X_s), \exists \omega' \in \mathcal{L}(X) \backslash \mathcal{L}_f(X_s)$ s.t. $P_{\Sigma_o}(\omega) = P_{\Sigma_o}(\omega')$ which is equivalent to asking if $P_{\Sigma_o}(\mathcal{L}(X) \cap \mathcal{L}_f(X_s)) \subseteq P_{\Sigma_o}(\mathcal{L}(X) \setminus \mathcal{L}_f(X_s))$.

Theorem 4 (Sufficient condition). *If $\mathcal{X}$-automata are closed under product, complementation, and projection, then the* LBTO *of $\mathcal{X}$-automata is decidable.*

Proof. For the proof, we provide a decision procedure for the LBTO of $\mathcal{X}$-automata if $\mathcal{X}$-automata are closed under product, complementation, and projection.

First, we transform X to an $\mathcal{X}$-automaton X' by labeling all locations in X as accepting locations. Thus, we have $\mathcal{L}(X) = \mathcal{L}_f(X')$. Since $\mathcal{X}$-automata are closed under complementation, we can build the complemented $\mathcal{X}$-automaton of X_s, denoted by $\overline{X_s}$. By the product operation, we can build two product $\mathcal{X}$-automata $Y_s = X' \times X_s$ and $Y_{ns} = X' \times \overline{X_s}$. Therefore, Y_s represents the secret part, i.e., $\mathcal{L}_f(Y_s) = \mathcal{L}(X) \cap \mathcal{L}_f(X_s)$, and Y_{ns} represents the non-secret part $\mathcal{L}_f(Y_{ns}) = \mathcal{L}(X) \setminus \mathcal{L}_f(X_s)$. Since $\mathcal{X}$-automata are closed under projection P_{Σ_o}, we can build two projection $\mathcal{X}$-automata $Y_s^{\Sigma_o}$ and $Y_{ns}^{\Sigma_o}$. We have $\mathcal{L}_f(Y_s^{\Sigma_o}) = P_{\Sigma_o}(\mathcal{L}_f(Y_s)) = P_{\Sigma_o}(\mathcal{L}(X) \cap \mathcal{L}_f(X_s))$ and $\mathcal{L}_f(Y_{ns}^{\Sigma_o}) = P_{\Sigma_o}(\mathcal{L}_f(Y_{ns})) = P_{\Sigma_o}(\mathcal{L}(X) \setminus \mathcal{L}_f(X_s))$. For checking if $\mathcal{L}_f(Y_s^{\Sigma_o}) \subseteq \mathcal{L}_f(Y_{ns}^{\Sigma_o})$, we build a product $\mathcal{X}$-automaton $Z = Y_s^{\Sigma_o} \times \overline{Y_{ns}^{\Sigma_o}}$ and check the emptiness problem of Z. If $\mathcal{L}_f(Z) = \emptyset$, then X is LBTO w.r.t X_s and Σ_o. As shown in [2], the emptiness problem of timed automata is decidable in PSPACE. Since $\mathcal{X}$ is a sub-class of timed automata, the emptiness problem of $\mathcal{X}$-automata is also decidable.

Therefore, the LBTO of $\mathcal{X}$-automata is decidable if $\mathcal{X}$-automata are closed under product, complementation, and projection. □

For instance, we check our sufficient condition on the subclasses mentioned in Sect. 2.2. According to [12], RTA satisfy the sufficient condition, and we know that the opacity of RTA is decidable [27,31]. However, ϵ-NTA and NTA are not closed under complementation. Although DTA and ERA are closed under complementation, they are not closed under projection. [11] shows that the opacity problems of ϵ-NTA, NTA, DTA, and ERA are undecidable.

Necessary Condition of CLTO. Given an $\mathcal{X}$-automaton X, and a secret subset of locations $Q_s \subseteq Q$, by Definition 5, the CLTO problem asks if $\forall \omega \in Tr_X(Q_0, Q_s), \exists \omega' \in Tr_X(Q_0, Q \setminus Q_s)$ *s.t.* $P_{\Sigma_o}(\omega) = P_{\Sigma_o}(\omega')$.

The following lemma states that the universality problem of $\mathcal{X}$-automata can be reduced to an equivalent CLTO problem of $\mathcal{X}$-automata.

Lemma 6. *Given an $\mathcal{X}$-automaton X, there exists an $\mathcal{X}$-automaton X' s.t. the universality problem of X is equivalent to the* `CLTO` *problem of X'.*

Proof. Given an $\mathcal{X}$-automaton $X = (\Sigma, Q, Q_0, Q_f, \mathcal{C}, \Delta)$, the universality problem asks if $\mathcal{L}_f(X) = (\Sigma \times \mathbb{R}_{\geq 0})^*$.

Similar to the proof of Lemma 2, we first introduce a new non-accepting location $\tilde{q}$ and then build its complete $\mathcal{X}$-automaton $X' = (\Sigma, \tilde{Q}, Q_0, Q_f, C, \Delta')$ with $\tilde{Q} = Q \cup \tilde{q}$, which satisfies $\mathcal{L}_f(X) = \mathcal{L}_f(X')$ and $\mathcal{L}(X') = Tr_{X'}(Q_0) = (\Sigma \times \mathbb{R}_{\geq 0})^*$.

Let the observable subset $\Sigma_o = \Sigma$ and the secret location subsets $Q_s = \tilde{Q} \setminus Q_f$. By Definition 5, the `CLTO` problem of X' w.r.t Q_s and Σ_o asks if

$$\forall \omega \in Tr_{X'}(Q_0, Q_s), \exists \omega' \in Tr_{X'}(Q_0, \tilde{Q} \setminus Q_s) \ s.t. \ P_\Sigma(\omega) = P_\Sigma(\omega')$$

which is equivalent to

$$\begin{aligned} &\forall \omega \in Tr_{X'}(Q_0), \exists \omega' \in Tr_{X'}(Q_0, \tilde{Q} \setminus Q_s) \ s.t. \ P_\Sigma(\omega) = P_\Sigma(\omega') \\ \Leftrightarrow &\forall \omega \in \mathcal{L}(\mathcal{A}'), \exists \omega' \in \mathcal{L}_f(X') \ s.t. \ P_\Sigma(\omega) = P_\Sigma(\omega') \\ \Leftrightarrow &P_\Sigma(\mathcal{L}(X') \subseteq P_\Sigma(\mathcal{L}_f(X')). \end{aligned}$$

By definition, for the same automaton, the recognized language is a subset of the generated language, then $P_\Sigma(\mathcal{L}_f(X')) \subseteq P_\Sigma(\mathcal{L}(X'))$. Therefore, it asks if $P_\Sigma(\mathcal{L}_f(X')) = P_\Sigma(\mathcal{L}(X'))$ which equals

$$\begin{aligned} &P_\Sigma(\mathcal{L}_f(X')) = (\Sigma \times \mathbb{R}_{\geq 0})^* \\ \Leftrightarrow &P_\Sigma(\mathcal{L}_f(X)) = (\Sigma \times \mathbb{R}_{\geq 0})^* \\ \Leftrightarrow &\mathcal{L}_f(X) = (\Sigma \times \mathbb{R}_{\geq 0})^* \end{aligned}$$

Therefore, it is equivalent to the universality problem of X. □

Theorem 5 (Necessary condition). *If the* `CLTO` *of $\mathcal{X}$-automata is decidable, then the universality problem of $\mathcal{X}$-automata is decidable.*

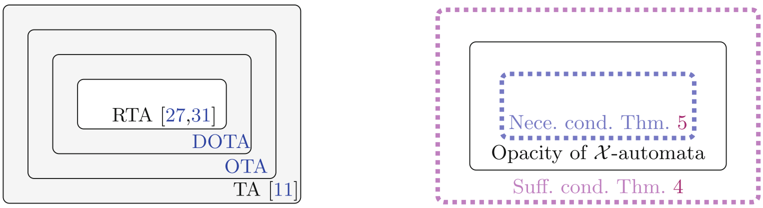

Fig. 4. Left: the decidability and undecidability results on the opacity of timed automata; **Right**: the sufficient condition and necessary condition for the decidability of the opacity of sub-class $\mathcal{X}$-automata.

5 Discussion and Conclusion

In this paper, we systematically examined three opacity problems (LBTO, ILTO, and CLTO) for TA with their transformations. We prove the undecidability of these opacity problems for one-clock timed automata, addressing a gap in prior work. Additionally, we provide a constructive proof confirming the decidability of opacity for TA under discrete-time semantics, offering a general verification algorithm. Finally, we propose a sufficient condition for LBTO and a necessary condition for CLTO, elucidating the system properties guiding the design of an opaque timed system.

In Fig. 4, the figure on the left side summarizes the decidability (for RTA) and undecidability (gray part in the figure) results on the opacity of different classes of timed automata; the figure on the right side illustrates the relation between the opacity problem, the necessary condition, and the sufficient condition. Hence, one question is if there exists a subclass $\mathcal{X}$-automata such that RTA $\subset$ $\mathcal{X}$-automata and the opacity of $\mathcal{X}$-automata is decidable. Another interesting question is whether we can find some tighter sufficient conditions and necessary conditions on the decidability of timed opacity or even a sufficient and necessary condition.

Acknowledgments. Thank the anonymous reviewers very much for their valuable suggestions. J. An, Q. Gao, and N. Zhan are partly supported by the NSFC under grants No. 62192732 and No. 62032024, and by the National Key R&D Program of China under grant No. 2022YFA1005101. J. An and I. Hasuo are supported by ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), JST.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Abdulla, P.A., Deneux, J., Ouaknine, J., Quaas, K., Worrell, J.: Universality analysis for one-clock timed automata. Fundam. Informaticae **89**(4), 419–450 (2008). http://content.iospress.com/articles/fundamenta-informaticae/fi89-4-04
2. Alur, R., Dill, D.L.: A theory of timed automata. Theor. Comput. Sci. **126**(2), 183–235 (1994). https://doi.org/10.1016/0304-3975(94)90010-8
3. Alur, R., Fix, L., Henzinger, T.A.: Event-clock automata: a determinizable class of timed automata. Theor. Comput. Sci. **211**(1–2), 253–273 (1999). https://doi.org/10.1016/S0304-3975(97)00173-4
4. Ammar, I., Touati, Y.E., Yeddes, M., Mullins, J.: Bounded opacity for timed systems. J. Inf. Secur. Appl. **61**, 102926:1–102926:13 (2021). https://doi.org/10.1016/j.jisa.2021.102926
5. André, É., Lime, D., Marinho, D., Sun, J.: Guaranteeing timed opacity using parametric timed model checking. ACM Trans. Softw. Eng. Methodol. **31**(4), 64:1–64:36 (2022). https://doi.org/10.1145/3502851
6. André, É., Sun, J.: Parametric timed model checking for guaranteeing timed opacity. In: Chen, Y.-F., Cheng, C.-H., Esparza, J. (eds.) ATVA 2019. LNCS, vol. 11781, pp. 115–130. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-31784-3_7

7. Badouel, É., Bednarczyk, M.A., Borzyszkowski, A.M., Caillaud, B., Darondeau, P.: Concurrent secrets. Discret. Event. Dyn. Syst. **17**(4), 425–446 (2007). https://doi.org/10.1007/s10626-007-0020-5
8. Bengtsson, J., Yi, W.: Timed automata: semantics, algorithms and tools. In: Desel, J., Reisig, W., Rozenberg, G. (eds.) ACPN 2003. LNCS, vol. 3098, pp. 87–124. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27755-2_3
9. Bérard, B., Gastin, P., Petit, A.: On the power of non-observable actions in timed automata. In: Puech, C., Reischuk, R. (eds.) STACS 1996. LNCS, vol. 1046, pp. 255–268. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-60922-9_22
10. Bortz, A., Boneh, D.: Exposing private information by timing web applications. In: Williamson, C.L., Zurko, M.E., Patel-Schneider, P.F., Shenoy, P.J. (eds.) WWW 2007, pp. 621–628. ACM (2007). https://doi.org/10.1145/1242572.1242656
11. Cassez, F.: The dark side of timed opacity. In: Park, J.H., Chen, H.-H., Atiquzzaman, M., Lee, C., Kim, T., Yeo, S.-S. (eds.) ISA 2009. LNCS, vol. 5576, pp. 21–30. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02617-1_3
12. Dima, C.: Real-time automata. J. Autom. Lang. Comb. **6**(1), 3–23 (2001). https://doi.org/10.25596/jalc-2001-003
13. Falcone, Y., Marchand, H.: Enforcement and validation (at runtime) of various notions of opacity. Discret. Event Dyn. Syst. **25**(4), 531–570 (2015). https://doi.org/10.1007/s10626-014-0196-4
14. Felten, E.W., Schneider, M.A.: Timing attacks on web privacy. In: Gritzalis, D., Jajodia, S., Samarati, P. (eds.) CCS 2000, pp. 25–32. ACM (2000). https://doi.org/10.1145/352600.352606
15. Gruber, H., Holzer, M., Kiehn, A., König, B.: On timed automata with discrete time – structural and language theoretical characterization. In: De Felice, C., Restivo, A. (eds.) DLT 2005. LNCS, vol. 3572, pp. 272–283. Springer, Heidelberg (2005). https://doi.org/10.1007/11505877_24
16. Han, X., Zhang, K., Li, Z.: Verification of strong k-step opacity for discrete-event systems. In: CDC 2022, pp. 4250–4255. IEEE (2022). https://doi.org/10.1109/CDC51059.2022.9993023
17. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages and Computation. Addison-Wesley (1979)
18. Jacob, R., Lesage, J., Faure, J.: Overview of discrete event systems opacity: models, validation, and quantification. Annu. Rev. Control. **41**, 135–146 (2016). https://doi.org/10.1016/j.arcontrol.2016.04.015
19. Jancar, J., et al.: "They're not that hard to mitigate": what cryptographic library developers think about timing attacks. In: S&P 2022, pp. 632–649. IEEE (2022). https://doi.org/10.1109/SP46214.2022.9833713
20. Lin, F.: Opacity of discrete event systems and its applications. Automatica **47**(3), 496–503 (2011). https://doi.org/10.1016/j.automatica.2011.01.002
21. Liu, S., Yin, X., Zamani, M.: On a notion of approximate opacity for discrete-time stochastic control systems. In: ACC 2020, pp. 5413–5418. IEEE (2020). https://doi.org/10.23919/ACC45564.2020.9147235
22. Ouaknine, J., Worrell, J.: Revisiting digitization, robustness, and decidability for timed automata. In: LICS 2003, pp. 198–207. IEEE Computer Society (2003). https://doi.org/10.1109/LICS.2003.1210059
23. Saboori, A., Hadjicostis, C.N.: Notions of security and opacity in discrete event systems. In: CDC 2007, pp. 5056–5061. IEEE (2007). https://doi.org/10.1109/CDC.2007.4434515

24. Saboori, A., Hadjicostis, C.N.: Verification of infinite-step opacity and complexity considerations. IEEE Trans. Autom. Control **57**(5), 1265–1269 (2012). https://doi.org/10.1109/TAC.2011.2173774
25. Saboori, A., Hadjicostis, C.N.: Verification of initial-state opacity in security applications of discrete event systems. Inf. Sci. **246**, 115–132 (2013). https://doi.org/10.1016/j.ins.2013.05.033
26. Wang, L., Zhan, N.: Decidability of the initial-state opacity of real-time automata. In: Jones, C., Wang, J., Zhan, N. (eds.) Symposium on Real-Time and Hybrid Systems. LNCS, vol. 11180, pp. 44–60. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01461-2_3
27. Wang, L., Zhan, N., An, J.: The opacity of real-time automata. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **37**(11), 2845–2856 (2018). https://doi.org/10.1109/TCAD.2018.2857363
28. Wu, Y., Lafortune, S.: Comparative analysis of related notions of opacity in centralized and coordinated architectures. Discret. Event Dyn. Syst. **23**(3), 307–339 (2013). https://doi.org/10.1007/s10626-012-0145-z
29. Yin, X., Lafortune, S.: A new approach for the verification of infinite-step and k-step opacity using two-way observers. Automatica **80**, 162–171 (2017). https://doi.org/10.1016/j.automatica.2017.02.037
30. Yin, X., Zamani, M., Liu, S.: On approximate opacity of cyber-physical systems. IEEE Trans. Autom. Control **66**(4), 1630–1645 (2021). https://doi.org/10.1109/TAC.2020.2998733
31. Zhang, K.: State-based opacity of real-time automata. In: Castillo-Ramirez, A., Guillon, P., Perrot, K. (eds.) 27th IFIP WG 1.5 International Workshop on Cellular Automata and Discrete Complex Systems, AUTOMATA 2021. OASIcs, vol. 90, pp. 12:1–12:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/OASIcs.AUTOMATA.2021.12