Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 58-1 (2024) 246–251

Observer Design for Timed Automata under No Observation

C. Gao * **D. Lefebvre** * **C. Seatzu** ** **Z. Li** *** **A. Giua** **

* *GREAH, Université Le Havre Normandie, Le Havre 76600, France, (e-mail: chao.gao@univ-lehavre.fr;dimitri.lefebvre@univ-lehavre.fr).*

** *DIEE, University of Cagliari, Cagliari 09124, Italy, (e-mail: carla.seatzu@unica.it;giua@unica.it).*

*** *School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China, (e-mail: zhwli@xidian.edu.cn).*

Abstract: This paper considers a class of timed discrete event systems (DESs) with a single clock. The timing structure is characterized by a timing function and a clock resetting function, where the former restricts transitions to occur when the clock takes a value in a given time interval, and the latter indicates how the clock value is updated upon the occurrence of transitions. Given a set of current discrete states in which the system can be, we assume that no information on the occurrence of events is captured (all such information is destroyed or lost), and only the clock that measures the time is reliable. We propose a state observer, in terms of a deterministic finite automaton, which enables us to compute the set of states in which the system can be at a certain time instant. The proposed observer is the basic step towards the construction of an observer that considers both unobservable and observable evolutions. Thus it can be considered as a first step in solving various problems related to partial observation of timed DESs such as opacity verification, fault diagnosis and diagnosability analysis.

Keywords: Discrete event system, automaton, unobservable evolution, observer.

1. INTRODUCTION

A discrete event system (DES) is an event-driven system where its state space is a discrete set and transitions from one discrete state to another are triggered by the occurrence of an event. The methodology employed in the context of DES can be extended to various real-world systems, including networking systems and batch processes utilized in manufacturing systems and chemical systems. In these systems, operations progress through distinct stages or batches, each of which can be characterized by a sequence of discrete events. Within the DESs community, a significant area of research revolves around partial observation. This is motivated by limited sensor availability for recording specific events, loss of system output measurements, and interruptions during communication and data transmission. In such instances, only a subset of events can be effectively measured. Researchers in the DESs community have dedicated to developing methods, tools and strategies based on partial observations, making it a fundamental and critical area of study.

In the context of dynamical systems, including logical and timed aspects, the notion of observer is a fundamental concept. In the domain of DESs, an observer is defined as a deterministic automaton that reconstructs with the utmost precision the set of current states consistent with an observation. It allows one to make informed decisions and ensures the robust and reliable operation of DESs. The notion of observer is pivotal to addressing a wide range of critical challenges in the field, including supervisory control, system diagnosis, and diagnosability.

Timed DESs introduce a specific timing structure characterised by a set of additional constraints governing the system's evolution. The concept of observers has been explored for several specific classes of timed DESs. For instance, Lai et al. (2019) design observers for a specific class of weighted automata, known as Max-plus automata. These automata are closely related to timed automata, particularly when timed interpretations are applied to their weights. Additionally, Zhang. (2021) contributes to the field by developing an observer for real-time automata, focused on estimating the current state based on timed output sequences. However, the proposed observer may not be unique due to various possible selections of the set of events under consideration.

Li et al. (2021) focus on designing observers for a class of single-clock timed DESs, where events occur at constant time instants. In contrast, Gao et al. (2023) and Lefebvre et al. (2023) focus on a class of timed automata, where transitions are associated with specific time intervals indicating when they may occur. However, these works deal with state estimation and fault diagnosis of a class of timed automata under a rather restrictive scenario, where the endowed single clock is reset to zero after each event occurrence. Gao et al. (2024) discuss a more general class of one-clock timed automata that do not necessarily require clock resetting at each event occurrence. An online approach is derived to recursively estimate the current discrete state as new observations are collected. State estima-

⋆ This work is partially supported by PRIN project MOTOWN: Motown: Smart Production Planning and Control for Manufacturing of Electric Vehicle Powertrain in Industry 4.0 Environment, Call 2022 Prot. 20228XEHRE, partially supported by BORSA DI RICERCA BANDO N. 34/2023 Metodi formali per il tracciamento su sistemi funded by the University of Cagliari, partially supported by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU, and partially supported by RIN research TREMPLIN 2020-2024 ARBRE funded by Université Le Havre Normandie and Région Normandie.

10.1016/j.ifacol.2024.07.042

tion of timed DESs has also been considered in the framework of time Petri nets in Basile et al. (2016), but no general approach concerning the construction of an observer exists.

Another parallel line of research is that of hybrid systems (HS). The timed automata proposed in Alur et al. (1994) are subclasses of HS models which provide a convenient framework for appropriately representing and efficiently reasoning about cyber-physical systems subject to real-time constraints. Related works such as Tripakis (2002) and Bouyer et al. (2021) embed state estimation into the diagnosis of timed automata. These works propose online diagnosers to keep track of all the possible discrete states and the associated clock constraints at each time an event is measured after a delay. Nevertheless, an approach to construct an offline observer that provides all possible state estimations that can be encountered in this context remains elusive, motivating us to explore observer design for timed automata as a self-standing problem.

In this paper, we assume the worst-case scenario where no logical information is captured (all such information is destroyed or lost), and only the clock that measures the time is reliable. This paper models such a system using the framework of one-clock timed DESs studied in Gao et al. (2024). The time structure is characterized by a timing function and a clock resetting function, where the former restricts transitions to occur when the clock takes a value in an associated time interval, and the latter indicates how the clock value is updated when a transition occurs. This paper provides an offline observer. The observer is a finite structure that permits the estimation of the state of a TFA as a function of the time that has elapsed since its initialization. Each observer state is a set of pairs; the first entry of which is a discrete state of the timed DES and the second entry is an interval of possible values for the clock.

The alphabet of the proposed observer consists of two events, 0+ and 1, which are clock events. Assume that an agent aiming to estimate the state of the timed DES initiates an observer timer, which is set to 0 with the start of the observation. Then, the observer is initialized. As time elapses, when the observer timer passes from an integer to a non-integer value, it generates an event 0+, when it passes from a non-integer to an integer value, it generates an event 1. The sequence of clock events, when executed in the observer, yields the state estimate.

This manuscript is organized as follows. Section 2 introduces the background of timed finite automata, and Section 3 details the problem statement. Section 4 recalls the notion of region automaton. Section 5 presents a method to design an observer for all possible unobservable evolutions emerging from a set of discrete states. Finally, Section 6 concludes this paper.

2. BASIC DEFINITIONS

Given an *alphabet* E representing a set of events, we denote by E^* the set of all finite strings on E, including the empty word ε. A string of events $w \in E^*$ is also called a *word* on E. The concatenation of two words $w_1 \in E^*$ and $w_2 \in E^*$ is a new word $w = w_1 \cdot w_2 \in E^*$ composed by the sequence of symbols in w_1 followed by the sequence of symbols in w_2.

A *nondeterministic finite automaton (NFA)* is a four-tuple $G_{nd} = (X, E, \Delta, X_0)$, where X and E are the sets of discrete states and events, respectively; $\Delta \subseteq X \times E \times X$ is the transition relation; $X_0 \subseteq X$ is the set of initial states, which may include more than one state in X. The transitive and reflexive closure of Δ is the relation $\Delta^* \subseteq X \times E^* \times X$ such that $(x, s, x') \in \Delta^*$ if the word s is generated from x and reaches x'. In particular, we have $(x, \varepsilon, x) \in \Delta^*$. In a *deterministic finite automaton (DFA)*, the transition function is defined in such a way that for each discrete state $x \in X$ and an enabled event $e \in E$, there is precisely one transition to another discrete state. This deterministic property ensures that the transition from one state to another is uniquely determined by the current state and the event.

By denoting the sets of non-negative real numbers and natural numbers as $\mathbb{R}_{\geq 0}$ and $\mathbb{N}$, respectively, the set of real numbers in $\mathbb{R}_{\geq 0}$ lying between a lower bound $m \in \mathbb{N}$ and an upper bound $n \in \mathbb{N} \cup \{+\infty\}$ is said to be a *time interval*. A *closed time interval* is denoted by $[m, n]$. In addition, an open segment (m, n) and semi-open segments $[m, n)$ or $(m, n]$ can also be *time intervals*. We denote the set of all time intervals and the set of all closed time intervals as $\mathbb{I}$ and $\mathbb{I}_c$, respectively, where $\mathbb{I}_c \subseteq \mathbb{I}$. Given two time intervals $I_1, I_2 \in \mathbb{I}$, we define their *addition* [1] as $I_1 \bigoplus I_2 = \{t_1 + t_2 \in \mathbb{R}_{\geq 0} \mid t_1 \in I_1, t_2 \in I_2\}$, and the *distance range* between them as $D(I_1, I_2) = \{|t_1 - t_2| \mid t_1 \in I_1, t_2 \in I_2\}$. For instance, given $I_1 = [0, 1)$ and $I_2 = [3, 4]$, it holds that $I_1 \bigoplus I_2 = [3, 5)$ and $D(I_1, I_2) = (2, 4]$.

Definition 1. A *timed finite automaton* (TFA) is a six-tuple $G = (X, E, \Delta, \Gamma, Reset, X_0)$ that operates under a single clock, where X is a finite set of discrete states, E is an alphabet, $\Delta \subseteq X \times E \times X$ is a transition relation, $\Gamma : \Delta \to \mathbb{I}_c$ is a timing function, $Reset : \Delta \to \mathbb{I}_c \cup \{id\}$ is a clock resetting function such that for $\delta \in \Delta$, the clock is reset to be an integer value in a time interval $I \in \mathbb{I}_c$ ($Reset(\delta) = I$), or the clock is not reset ($Reset(\delta) = id$), and $X_0 \subseteq X$ is the set of initial discrete states.$\diamond$

For simplicity, we assume the initial clock value is set to 0. A transition $(x, e, x') \in \Delta$ signifies that the occurrence of event $e \in E$ leads to a state transition from x to x' in the state space X. The time interval $\Gamma((x, e, x'))$ specifies the range of clock values during which the event e may occur, and $Reset((x, e, x')) \in \mathbb{I}_c$ denotes the range of values to which the clock is reset, with $Reset((x, e, x')) = id$ indicating no clock reset.

The *set of output transitions at state* x is defined as $Out(x) = \{(x, e, x') \in \Delta \mid e \in E, x' \in X\}$, and *the set of input transitions at* x is defined as $In(x) = \{(x', e, x) \in \Delta \mid e \in E, x' \in X\}$. We assume that a TFA operates under weak time semantics that permits a system to persist in any discrete state indefinitely. A *timed state* is defined as a pair $(x, \theta) \in X \times \mathbb{R}_{\geq 0}$, where $\theta \in \mathbb{R}_{\geq 0}$ is the current value of the clock. In other words, a timed state (x, θ) keeps track of the current clock assignment θ while G stays at state x. The behaviour of a TFA is described via its timed runs. A *timed run* ρ of length $k \geq 0$ from $t_0 \in \mathbb{R}_{\geq 0}$ to $t_k \in \mathbb{R}_{\geq 0}$ is a sequence of $k+1$ timed states $(x_{(i)}, \theta_{(i)}) \in X \times \mathbb{R}_{\geq 0}$ $(i = 0, \cdots, k)$, and k pairs $(e_i, t_i) \in E \times \mathbb{R}_{\geq 0}$ $(i = 1, \cdots, k)$, represented as

$$\rho : (x_{(0)}, \theta_{(0)}) \xrightarrow{(e_1, t_1)} \cdots (x_{(k-1)}, \theta_{(k-1)}) \xrightarrow{(e_k, t_k)} (x_{(k)}, \theta_{(k)}) \quad (1)$$

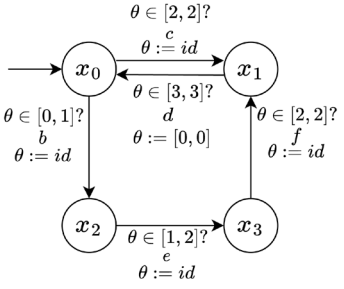
such that $(x_{(i-1)},\ e_i,\ x_{(i)}) \in \Delta$, $t_{i-1} \leq t_i$ and the following conditions hold for all $i = 1, \cdots, k$:

- $\theta_{(i)} \in Reset((x_{(i-1)},\ e_i,\ x_{(i)}))$ and $\theta_{(i-1)} + t_i - t_{i-1} \in \Gamma((x_{(i-1)}, e_i, x_{(i)}))$, if $Reset((x_{(i-1)}, e_i, x_{(i)})) \neq id$;
- $\theta_{(i)} = \theta_{(i-1)} + t_i - t_{i-1} \in \Gamma((x_{(i-1)}, e_i, x_{(i)}))$, if $Reset((x_{(i-1)}, e_i, x_{(i)})) = id$.

[1] The addition operation is associative and commutative and can be extended to $n > 2$ time intervals $\bigoplus_{i=1}^{n} I_i = I_1 \bigoplus \cdots \bigoplus I_n$.

We define the *timed word generated by* ρ as $\sigma(\rho) = (e_1, t_1)(e_2, t_2)\cdots(e_k, t_k) \in (E \times \mathbb{R}_{\geq 0})^*$, the *logical word generated by* ρ as $S(\sigma(\rho)) = e_1 e_2 \cdots e_k$ via a function defined as $S : (E \times \mathbb{R}_{\geq 0})^* \to E^*$. For the timed run of length 0 as $\rho : (x_{(0)}, \theta_{(0)})$, we have $S(\sigma(\rho)) = \varepsilon$ and $\sigma(\rho) = \lambda$, where λ denotes the *empty timed word* in $E \times \mathbb{R}_{\geq 0}$. For the timed word $\sigma(\rho)$ generated from an arbitrary timed run ρ, it is $\lambda \cdot \sigma(\rho) = \sigma(\rho) = \sigma(\rho) \cdot \lambda$. The *starting discrete state* and the *ending discrete state* of a timed run ρ are denoted by $x_{st}(\rho) = x_{(0)}$ and $x_{en}(\rho) = x_{(k)}$, respectively. The *starting time* and the *ending time* of ρ are denoted by $t_{st}(\rho) = t_0$ and $t_{en}(\rho) = t_k$, respectively. In addition, the *duration of* ρ is denoted as $T(\rho) = t_k - t_0$. The set of timed runs generated by G is denoted as $\mathscr{R}(G)$. A *timed evolution* of G from time 0 to $t \in \mathbb{R}_{\geq 0}$ is defined by a pair $(\sigma(\rho), t) \in (E \times \mathbb{R}_{\geq 0})^* \times \mathbb{R}_{\geq 0}$, where $t_{en}(\rho) \leq t$. Note that $t - t_{en}(\rho)$ is the time that the system stays at the ending discrete state $x_{en}(\rho)$.

Example 1. Given a TFA $G = (X, E, \Delta, \Gamma, Reset, X_0)$ in Fig. 1(a) with $X = \{x_0, x_1, x_2, x_3\}$, and $E = \{b, c, d, e, f\}$, the initial state in $X_0 = \{x_0\}$ is marked by an input arrow. The information given by the timing function Γ and the clock resetting function *Reset* defined in Fig. 1(b) is presented on the edges. Given an edge denoting a transition $\delta \in \Delta$, the label $\theta \in \Gamma(\delta)$? on the edge specifies if δ is enabled with respect to θ; the label $\theta :\in Reset(\delta)$ (resp., $\theta := id$) on the edge specifies to which range θ belongs (resp., specifies that the clock is not reset) after the transition is fired. Consider a timed run $\rho : (x_0, 0) \xrightarrow{(c,2)} (x_1, 2) \xrightarrow{(d,3)} (x_0, 0)$ that starts from $x_{st}(\rho) = x_0$ at $t_{st}(\rho) = 0$ and terminates in $x_{en}(\rho) = x_0$ at $t_{en}(\rho) = 3$. Two transitions (x_0, c, x_1), and (x_1, d, x_0) occur at time instants $t_1 = 2$, and $t_2 = 3$, respectively. ◇



(a) A TFA G.

$\delta \in \Delta$	$\Gamma(\delta)$	$Reset(\delta)$
(x_0, c, x_1)	$[2,2]$	id
(x_0, b, x_2)	$[0,1]$	id
(x_1, d, x_0)	$[3,3]$	$[0,0]$
(x_2, e, x_3)	$[1,2]$	id
(x_3, f, x_1)	$[2,2]$	id

(b) Timing function and clock resetting function.

Fig. 1. A TFA G w.r.t. the given timing function and clock resetting function.

3. PROBLEM STATEMENT

In this paper, we focus on a state estimation problem. In particular, we consider scenarios where all measurements are lost, e.g., due to sensor failures or communication interruptions. Thus, we assume that all events in E are unobservable. Given a set of initial states X_0, we design an observer, implemented as a DFA, which provides the set of discrete states in which the system can be at any time instant t, referred as the state estimation at t.

Definition 2. Given a TFA $G = (X, E, \Delta, \Gamma, Reset, X_0)$, $x' \in X$ is said to be a *reachable state* from $x \in X$ at $T \in \mathbb{R}_{\geq 0}$ if there exists a timed evolution $(\sigma(\rho), t) \in (E \times \mathbb{R}_{\geq 0})^* \times \mathbb{R}_{\geq 0}$ of G such that $t - t_{st}(\rho) = T$, $x_{st}(\rho) = x$, and $x_{en}(\rho) = x'$. The set of all reachable states from x at T is denoted as $UR(x, T)$. ◇

In simple words, if there exists a timed evolution that leads the system from x to x' with an elapsed time T, then x' is reachable from x at time T. Given $X_0 \subseteq X$, the proposed observer aims at providing the reachable discrete states from all $x \in X_0$ at all time instants $T \geq 0$.

4. REGION AUTOMATON

Alur et al. (1994) propose the notion of *region automaton*, which represents the behavior of a timed automaton over regions of the time domain. In this section, we first recall the notion of *region automaton* in Gao et al. (2020), which provides a discrete event description of a given TFA. Then, we formalize the evolution of a region automaton utilizing the notion of NFA.

Definition 3. Let m_x (resp., M_x) be the minimal (resp., maximal) integer in $\{\Gamma(\delta) | \delta \in Out(x) \vee (\delta \in In(x), Reset(\delta) = id)\} \cup \{Reset(\delta) \neq id | \delta \in In(x)\}$. The *regions of a discrete state* x is defined as $R(x) = \{[m_x, m_x], (m_x, m_x + 1), \cdots, (M_x - 1, M_x), [M_x, M_x], (M_x, +\infty)\}$, and the initial region of x is defined as $r_{ini}(x) = [m_x, m_x]$. The successor region of $r \in R(x) \setminus \{(M_x, +\infty)\}$ is defined as

$$succ(r) = \begin{cases} [j, j] & \text{if } r = (j-1, j); \\ (j-1, j) & \text{if } r = [j-1, j-1] \neq [M_x, M_x], \\ (M_x, +\infty) & \text{if } r = [M_x, M_x]. \end{cases}$$

The integer m_x (resp., M_x) represents the minimal (resp., maximal) clock value that can enable an output transition at x and that can reach x by an input transition. The set of regions $R(x)$ partitions the clock values in $[m_x, +\infty)$ at x into the integer points belonging to the interval $[m_x, M_x]$, the open segments between them, and an interval $[M_x, +\infty)$.

Definition 4. (Gao et al. (2020)). Given a TFA $G = (X, E, \Delta, \Gamma, Reset, X_0)$, the *region automaton* of G is an NFA $RA(G) = (V, E_r, \Delta_r, V_0)$, where

- $V \subseteq X \times \bigcup_{x \in X} R(x)$ is the finite set of extended states, where each extended state is a pair (x, r) with $x \in X$ and $r \in R(x)$;
- $E_r \subseteq E \cup \{\tau\}$ is the alphabet, where the event τ implies time elapsing from any clock value $\theta \in r$ to any $\theta' \in succ(r)$ when G stays at $x \in X$;
- $\Delta_r \subseteq V \times E_r \times V$ is the transition relation, where the transitions in Δ_r are defined by the following rules:
  - $((x, r), \tau, (x, succ(r))) \in \Delta_r$ if $r, succ(r) \in R(x)$; this corresponds to a time-driven evolution of G from a clock value in r to another clock value in $succ(r)$ while G is at x;
  - $((x, r), e, (x', r')) \in \Delta_r$ if $(x, e, x') \in \Delta$, $r \subseteq \Gamma((x, e, x'))$, and one of the following conditions holds: (a) $Reset((x, e, x')) \neq id$, $r' \in Reset((x, e, x'))$, namely the clock is reset after (x, e, x') occurs; (b) $Reset((x, e, x')) = id$, $r' = r$, namely the clock is not reset after (x, e, x') occurs. This indicates that the occurrence of event e yields extended state (x', r') when the current state of the system is x and the current clock is in r.
- $V_0 = \{(x, [0, 0]) \mid x \in X_0\} \subseteq V$ is the set of initial states.

Given $v = (x, r) \in V$, we further define a function $f_X : V \to X$ (resp., $f_R : V \to \bigcup_{x \in X} R(x)$) that maps an extended state in V to a discrete state $f_X(v) = x$ (resp., a region of the associated discrete state $f_R(v) = r$). ◇

Consider a timed run ρ of G with the form of Equation (1). The region automaton describes both the time-elapsed evolution and

the event-driven evolution in a discrete way via a string $s_R = e_0 \cdot s_{\tau 0} \cdots e_k \cdot s_{\tau k} \in E_r^*$ such that the following conditions hold by denoting $e_0 = \varepsilon$:

- $((x_{(i)}, r_i), s_{\tau i}, (x_{(i)}, r'_i)) \in \Delta_r^*$ and $s_{\tau i} \in \{\tau\}^*$ for $i = 0, \cdots, k$;
- $((x_{(i-1)}, r'_{i-1})), e_i, (x_{(i)}, r_i)) \in \Delta_r$, $\theta_{(i-1)} + t_i - t_{i-1} \in r'_{i-1} \subseteq \Gamma((x_{(i-1)}, e_i, x_i))$, and one of the following conditions holds: (a) $\theta_{(i)} \in r_i \subseteq Reset((x_{(i-1)}, e_i, x_{(i)})) \neq id$; (b) $\theta_{(i)} = \theta_{(i-1)} + t_i - t_{i-1} \in r'_{i-1} = r_i \subseteq \Gamma((x_{(i-1)}, e_i, x_{(i)}))$, $Reset((x_{(i-1)}, e_i, x_{(i)})) = id$.

In simple words, $s_{\tau i}$ involves only event τ and states $(x_{(i)}, r)$, where $r \in R(x_{(i)})$. It essentially represents the time elapsing in a discrete way while G is at $x_{(i)}$. The transition $((x_{(i-1)}, r'_{(i-1)}), e_i, (x_{(i)}, r_{(i)})) \in \Delta_r$ for $i = 1, \cdots, k$ implies event-driven evolution of G because of the occurrence of e_i at t_i. Based on that, we define the *duration range* of s_R as $d(s_R) = \bigoplus_{i=0}^{k} D(r_{(i)}, r'_{(i)})$.

Example 2. Given the TFA G in Fig. 1, the region automaton $RA(G)$ is depicted in Fig. 2. Each state of $RA(G)$ indicates a discrete state and a time interval to which G may belong. The transitions labelled with event τ imply only time elapses without the evolution of discrete states. The transitions labelled with an event in E imply discrete state evolution of G.

Continue with Example 1 and the timed evolution $(\sigma(\rho), 3)$. In $RA(G)$, there is a string s_R which can represent the time evolution and state evolution associated with $(\sigma(\rho), 3)$ in a discrete way, highlighted in red in Fig. 2, such that $((x_0, [0,0]), s_R, (x_1, [0,0]))) \in \Delta_r^*$, where $s_R = \tau\tau\tau\tau c \tau\tau d$. In details,

- $((x_0, [0,0]), \tau\tau\tau\tau c, (x_1, [2,2]))$ corresponds to the timed evolution from $(x_0, 0)$ to $(x_1, 2)$ by a pair $(c, 2)$; the duration equals to the time interval $[2,2]$ implying the elapsed time $t \in [2,2]$ at x_0 before c occurs;
- $((x_1, [2,2]), \tau\tau d, (x_0, [0,0]))$ corresponds to the timed evolution from $(x_1, 2)$ to $(x_0, 0)$ by a pair $(d, 3)$; the duration equals to the time interval $[1,1]$ implying the elapsed time $t \in [1,1]$ at x_1 before d occurs.

The duration of s_R can be computed as $[2,2] \bigoplus [1,1] = [3,3]$, implying the range of the total elapsed time. ◇

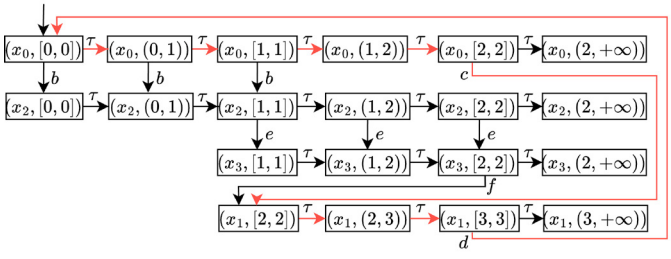


Fig. 2. Region automaton $RA(G)$.

5. OBSERVER DESIGN

In this section, we present the design of an observer in the form of a DFA. Suppose that an agent endeavors to estimate the state of a TFA without observing any sensor label. The agent initiates a clock for the observer, which starts at 0 when the observation begins. As time elapses, the observer timer generates an event 0+ when the observer timer transits from an integer to a non-integer value, and it generates an event 1 when the observer timer transits from a non-integer to an integer value. The execution of this event sequence in the observer produces the state estimates. We provide a formal algorithm to construct such an observer. Given an elapsed time with no observable event being measured starting from a set of discrete states, all the possible reached states can be inferred by analysing the reachability of the proposed observer.

We first introduce a *time string function* $TS : \mathbb{R}_{\geq 0} \to \{0+, 1\}^*$ to show how the elapsed time t recorded by the agent can be associated with a logical string $TS(t)$ defined by

$$TS(t) = \begin{cases} (0^+ \cdot 1)^{\lfloor t \rfloor} & \text{if } \lfloor t \rfloor = t; \\ (0^+ \cdot 1)^{\lfloor t \rfloor} \cdot 0^+ & \text{if } \lfloor t \rfloor < t, \end{cases}$$

where $\lfloor t \rfloor$ denotes the floor of t. In other words, the observer produces strings composed of alternating logical events 0+ and 1. For instance, given a measured elapsed time $0 < t_1 < 1$, the observer string denoted as $TS(t_1) = 0+$ yields the corresponding state estimation. Similarly, another string $TS(t_2) = (0+) \cdot 1$ leads to the state estimates corresponding to an elapsed time $t_2 = 1$, and so forth. We next propose the definition of the proposed observer.

Definition 5. Given a TFA $G = (X, E, \Delta, \Gamma, Reset, X_0)$ and its region automaton $RA(G) = (V, E_r, \Delta_r, V_0)$, an *observer for unobservable evolutions* from X_0 is defined as a DFA $Obs(G, X_0) = (Q, \{0+, 1\}, \delta_{obs}, q_0)$, where

- $Q \subseteq 2^V$ is the set of observer states, each of which is a set of extended states in V;
- $\{1, 0+\}$ is the set of clock events;
- $\delta_{obs} : Q \times \{0+, 1\} \to Q$ is the transition function. It is $\delta_{obs}^*(q_0, TS(t)) = q$ if and only if
$$(\forall v \in q_0)(\exists v' \in q)(v, s_R, v') \in \Delta_r, t \in d(s_R);$$
that is to say, the word $TS(t)$ on the alphabet $\{0+, 1\}$ leads from the initial state q_0 to state q, which is a set of extended states of $RA(G)$ that can be reached by a string s_R implying elapsed time t;
- $q_0 = \{(x, r_{ini}(x)) \mid x \in X_0\}$ is the initial state. ◇

Given a region automaton $RA(G) = (V, E_r, \Delta_r, V_0)$ and a time interval $I \in \{[0,0], (0,1), [1,1], \cdots\}$, we denote the *estimation from $v \in V$ at a time instant $t \in I$ with unobservable evolutions* as $EST(v, I) = \{v' \in V \mid (\exists s_R \in E_r^*)(v, s_R, v') \in \Delta_r^*, d(s_R) \cap I \neq \emptyset\}$. In simple words, the estimation comprises the extended states in V reachable from v through sequences with a duration falling within the range I. Next, we prove that the set of reachable states from x at time t, denoted as $UR(x, t)$, aligns with the observer state reached by a string $TS(t)$.

Theorem 1. Consider a TFA $G = (X, E, \Delta, \Gamma, Reset, X_0)$ and its observer $Obs(G, X_0) = (Q, \{0+, 1\}, \delta_{obs}, q_0)$. There exists an evolution (q_0, s_{obs}, q) in $Obs(G, X_0)$ if and only if there exists $t \in \mathbb{R}_{\geq 0}$ such that $TS(t) = s_{obs}$ and $f_X(q) = \bigcup_{x \in X_0} UR(x, t)$.

Proof. (if) Let $x, x' \in X$ and $t \in \mathbb{R}_{\geq 0}$ satisfy that $x' = \bigcup_{x \in X_0} UR(x, t)$. It can be inferred that there exists a timed evolution $(\sigma(\rho), t)$ from (x, θ) to (x', θ') starting from time 0. Let $RA(G) = (V, E_r, \Delta_r, V_0)$ be the region automaton of G. Then there exists a string s_R in $RA(G)$ such that $((x, r), s_R, (x', r')) \in \Delta_r$, where $\theta \in z$ and $\theta' \in z'$. Accordingly, there exists $I \in \{(0,1), [1,1], \cdots\}$ such that $t \in I$ and $(x', r') \in EST((x, r), I)$.

Therefore, a transition $(q_0, s_{obs}, (x', r'))$ exists in $Obs(G, X_0)$, and the sufficient condition holds.

(only if) Let (q_0, s_{obs}, q) be a transition of $Obs(G, X_0)$ such that $s_{obs} = TS(t)$. Then, there exists $I \in \{(0,1), [1,1], \cdots\}$ such that $t \in I$ and $q = \bigcup_{v \in q_0} EST(v, I)$. Then, in the region automaton $RA(G) = (V, E_r, \Delta_r, V_0)$, there exists a transition $(q_0, s_R, q) \in \Delta_r$ such that $t \in d(s_R)$. The transition (q_0, s_R, q) implies a timed evolution $(\sigma(\rho), t)$ produced by G. Therefore $f_X(q) = \bigcup_{x \in X_0} UR(x, t)$ holds. □

By defining *the successor region* for each $I \in \{[0,0], (0,1), [1,1], \cdots\}$ as

$$\phi(I) = \begin{cases} [j, j] & \text{if } r = (j-1, j); \\ (j-1, j) & \text{if } r = [j-1, j-1], \end{cases}$$

where $j \in \mathbb{N}_{\geq 0}$, we present an approach to constructing the above observer based on an analysis of the reachability of the region automaton $RA(G)$. In particular, we propose Algorithm 1 for constructing the observer $Obs(G) = (Q, \{0+, 1\}, \delta_{obs}, q_0)$ for unobservable evolutions from a given set of discrete states X_0. The algorithm follows these steps:

(1) Initialization: initialize the set of states Q as $\{q_0\}$, where q_0 represents the union of estimations from each $(x, r_{ini}(x))$ at 0 for $x \in X_0$; initialize δ_{obs} as an empty set; designate the index state of the observer as q, which is set to be q_0 initially, designate the interval I as $[0,0]$, and the end test STOP as false, i.e., 0.

(2) The *while* loop updates the states and transitions of the observer until STOP becomes true, i.e., 1. For each loop, we update I as the successor region $\phi(I)$, and denote the estimation from $v \in q_0$ at a time instant $t \in I$ as $\bar{q}$. If I is one of the open segments $(0,1)$, $(1,2)$, $\cdots$, we update δ_{obs} by including the transition $(q, 0+, \bar{q})$; if I equals to an integer value as $[1,1]$, $[2,2]$, $\cdots$, we update δ_{obs} by including the transition $(q, 1, \bar{q})$. The former implies that the observer state evolves from q to $\bar{q}$ after the time elapses more than 0 from the previous measurement, and the latter implies that the observer state evolves from q to $\bar{q}$ after the elapsed time reaches 1. After implementing the transition, we check if $\bar{q}$ is in Q or not: if not, we update Q by including $\bar{q}$ and update q by $\bar{q}$. If $\bar{q} \in Q$, which implies that all states and transitions have been implemented, the algorithm designates the end test STOP to become true, and stops the *while* loop. At the end, the algorithm returns the observer.

In summary, this algorithm constructs an observer that captures information for any given time instant $t \geq 0$ with no event observation. Observe that the number of extended states is finite, and the algorithm returns a structure in finite steps. Notice also that the observer obtained by Algorithm 1 is consistent with Definition 5. Specifically, the initial state q_0 of the observer provides the estimation for an immediate time $t \in [0,0]$ without any observation. The reachable states from $x \in X_0$ at $t \in (0,1)$ are equivalent to the discrete states $f_X(q_1)$, where q_1 is an observer state reached by an event $0+$ from q_0. Then, the transition with 1 from q_1 leads to an observer state, from which the reachable states from $x \in X_0$ at $t \in [1,1]$ can be inferred. Each string generated by the observer, such as $0+$, $0+\cdot 1$, $0+\cdot 1\cdot 0+$, etc., is composed of alternating clock events $0+$ and 1. These correspond to time instants in the respective partitions: $[0,0]$, $(0,1)$, $[1,1]$, etc., during which no observations are received.

Algorithm 1: Constructing an observer for unobservable evolutions of a TFA

Input: A region automaton $RA(G) = (V, E_r, \Delta_r, V_0)$, and a set of discrete states $X_0 = \bigcup_{v \in V_0} f_X(v)$

Output: Observer $Obs(G, X_0) = (Q, \{0+, 1\}, \delta_{obs}, q_0)$

```
let q_0 ← ⋃_{x∈X_0} EST((x, r_ini(x)), [0,0]), Q ← {q_0}
let δ_obs ← ∅, q ← q_0, I ← [0,0], and STOP ← 0
while STOP = 0 do
    let I ← φ(I) and q̄ ← ⋃_{v∈q_0} {EST(v, I)}
    if I = {(0,1), (1,2), ···} then
        δ_obs ← δ_obs ∪ {(q, 0+, q̄)}
    else
        δ_obs ← δ_obs ∪ {(q, 1, q̄)}
    if q̄ ∉ Q then
        let Q ← Q ∪ {q̄} and q ← q̄
    else
        let STOP ← 1
    return Obs(G) ← (Q, {0+, 1}, δ_obs, q_0)
```

The state estimation for no observation at t can be inferred by an observer state reached by a corresponding string $TS(t)$.

In the subsequent discussion, we provide an illustrative example to demonstrate how the proposed observer offers estimations for unobservable evolutions. In the example, we also highlight an intuitive process of obtaining a simplified observer from the proposed observer. However, due to space constraints, the algorithm for constructing the simplified observer is omitted. The simplified observer can be utilized to maintain the key aspects of discrete state estimation without delving into the specifics of clock values.

Example 3. Consider the TFA G in Fig. 1 and its region automaton $RA(G)$ in Fig. 2. Suppose that no further observations are recorded after the system moves to state $X_0 = \{x_0\}$. The observer $Obs(G, X_0)$ designed for unobservable evolutions is illustrated in Fig. 3. Each state of the observer corresponds to a set of extended states in $RA(G)$, providing estimations for discrete states and a range of potential clock values. The initial observer state, denoted as $q_0 = \{(x_0, [0,0]), (x_2, [0,0])\}$, provides the estimation at time 0 with no observation at states x_0 and x_2, both having a clock value equal to 0. Starting from q_0, the string labeled with $0+$ (or $0+\cdot 1$) signifies an elapsed time $t \in (0,1)$ (or $t \in [1,1]$). Subsequently, the transition labeled with $0+$ from q_0 leads to $q_1 = \bigcup_{v \in q_0} EST(v, (0,1)) = \{(x_0, (0,1)), (x_2, (0,1))\}$, while the transition labeled with 1 from q_1 leads to $q_2 = \bigcup_{v \in q_0} EST(v, [1,1]]) = \{(x_0, [1,1]), (x_2, [1,1]), (x_3, [1,1])\}$, respectively. Consider the scenario where no observation is recorded for $t = 6.5$. The observer $Obs(G, X_0)$ in Fig. 3 provides an estimation represented by $\{(x_0, (0,1)), (x_0, (2, +\infty)), (x_1, (3, +\infty)), (x_2, (0,1)), (x_2, (2, +\infty)), (x_3, (2, +\infty))\}$, derived from the reached observer state by string $s_{obs} = 0+\cdot 1\cdot 0+\cdot 1\cdot 0+\cdot 1\cdot 0+\cdot 1\cdot 0+\cdot 1\cdot 0+\cdot 1\cdot 0+$, implying an elapsed time $t \in (6,7)$.

In comparison to the observer $Obs(G, X_0)$, whose states are collections of extended states from $RA(G)$ that include clock information, a simplified observer depicted in Fig. 4, can be obtained by filtering out the clock information associated with

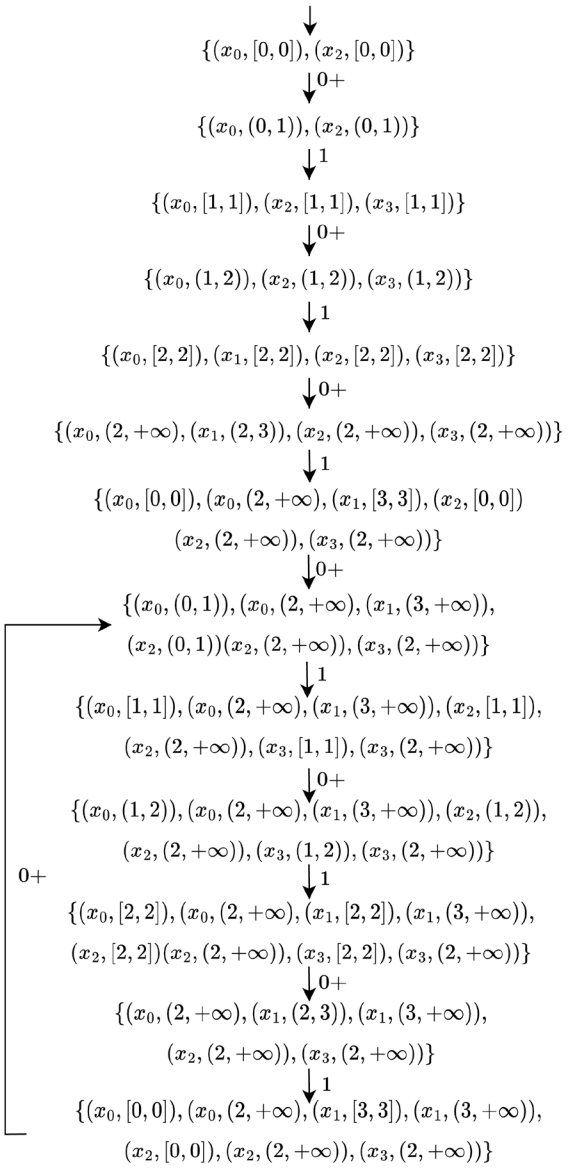


Fig. 3. Observer $Obs(G,X_0)$.

the states of G and merging identical estimations for discrete states. It basically provides the following information:

- The observer begins in the state $\{x_0,x_2\}$.
- In the absence of observations at $t=1$, the discrete state estimation expands to $\{x_0,x_2,x_3\}$.
- In the absence of observations at $t=2$, the observer updates the estimation of discrete states as $\{x_0,x_1,x_2,x_3\}$. $\diamond$

Remark 1. Note that, due to the considered semantics where a system can indefinitely remain in a discrete state, the set of consistent discrete states is non-decreasing as time elapses when no observation is collected.

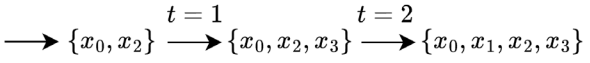


Fig. 4. A simplified observer, each state of which is a set of discrete states.

6. CONCLUSIONS

In this paper, we consider timed DESs equipped with a single clock. The timing structure is characterized by a timing function and a clock resetting function. Focusing on a designated set of discrete states as initial states, our contribution lies in the proposal of an observer that tracks elapsed time instants. Each state of the observer is a set of states of the associated region automaton, implying the possible current discrete states and the clock range. The proposed observer is implemented as a deterministic finite automaton, offering a further possibility for developing other properties from the aspect of discrete event systems. The proposed observer can be applied to real-world challenges. For instance, it can be employed in scenarios where no observations are measured for a certain duration due to interruptions or the loss of output measurements from a system. For future work, we are interested in the opacity of timed DES based on the proposed observer. In addition, we aim to develop an observer that provides accurate clock estimation.

REFERENCES

A. Lai, S. Lahaye, and A. Giua. State estimation of max-plus automata with unobservable events. *Automatica*, 105: 36 – 42, 2019.

K. Zhang. State-based opacity of real-time automata. *27th IFIP WG 1.5 Int. Workshop on Cellular Automata and Discrete Complex Systems (AUTOMATA 2021)*, 12: 1–15, 2021.

J. Li, D. Lefebvre, and C. N. Hadjicostis. Observers for a class of timed automata based on elapsed time graphs. *IEEE Trans. on Automatic Control*, 67(2) : 767–779, 2021.

F. Basile, M. P. Cabasino, and C. Seatzu. Diagnosability analysis of labeled time Petri net systems. *IEEE Trans. on Automatic Control*, 62(3) : 1384–1396, 2016.

D. Lefebvre, Z. Li, and Y. Liang. Diagnosis of timed patterns for discrete event systems by means of state isolation. *Automatica*, 153: 111045, 2023.

C. Gao, D. Lefebvre, C. Seatzu, Z. Li, and A. Giua. Fault Diagnosis of Timed Discrete Event Systems. In *Proc. of IFAC-PapersOnLine*, 56(2): 9612-9617, 2023.

C. Gao, D. Lefebvre, C. Seatzu, Z. Li, and A. Giua. A region-based approach for state estimation of timed automata under no event observation. In *Proc. of IEEE Int. Conf. on Emerging Technologies and Factory Automation*, volume 1, pages 799–804. IEEE, 2020.

R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

S. Tripakis. Fault diagnosis for timed automata. In *Proc. of the 7th Int. Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems: Co-sponsored by IFIP WG 2.2*, pages 205–224, 2002.

P. Bouyer, L. Henry, S. Jaziri, T. Jéron and N. Markey. Diagnosing timed automata using timed markings. *Int. Journal on Software Tools for Technology Transfer*, 23: 229–253, 2021.

C. Gao, D. Lefebvre, C. Seatzu, Z. Li and A. Giua. State estimation of timed automata under partial observation. Conditionally accepted by *IEEE Trans. on Automatic Control*, 2024.