State estimation of max-plus automata with unobservable events

Aiwen Lai, Sébastien Lahaye, Alessandro Giua

State Estimation of Max-Plus Automata with Unobservable Events ⋆

Aiwen Lai [a], Sébastien Lahaye [a,⋆], Alessandro Giua [b]

[a] *Laboratoire Angevin de Recherche en Ingénierie des Systèmes, Université d'Angers, 49000 Angers, France*

[b] *Department of Electrical and Electronic Engineering, University of Cagliari, 09123 Cagliari, Italy*

Abstract

The state estimation problem is a fundamental issue in discrete event systems. Partial observations arise when the occurrence of some events cannot be detected. The considered problem then consists in finding all the states in which the system may be when an observed sequence is given. To the best of our knowledge there are few works dealing with this problem in the framework of timed discrete event systems. In this paper we investigate state estimation for systems represented as max-plus automata. Max-plus automata represent a particular class of weighted automata and if a timed interpretation is given to weights, then max-plus automata are strongly related to timed automata. We first give the definition of consistent states with respect to an observed timed sequence and a given time instant. Then, based on the state vectors of a max-plus automaton, an algorithm is proposed to compute the set of all consistent states.

Key words: Discrete event systems; max-plus automata; partial observation; state estimation.

1 Introduction

The state estimation of a dynamic system is a fundamental issue in control theory. Two relevant properties, i.e., observability and detectability, are introduced to investigate this problem for time driven systems (TDS) and discrete event systems (DES). In TDS, the state estimation aims at generating an estimate $\hat{x}(t)$ of the current state $x(t)$ at a time instant t while in DES, it consists in characterizing all possible states with respect to a finite set of observed outputs [9].

The state estimation plays a very important role in some applications of DES. One related application is opacity. A system is said to be current state opaque if the intruder can not decide if the current state surely belongs to a set of states. Another important application is fault diagnosis. The failures are often associated with some unobservable events and diagnosis aims at determining the occurrence of a fault event [18]. In [25], the failures are modeled by a particular subset of states and the diagnosis problem then reduces to a state estimation problem. Automata and Petri nets (PNs) have been intensively used to deal with the state estimation problem over the past few decades.

The embryonic form of state estimation problem first appeared in [23]. Since then, this problem has been studied by many researchers. Ramadge [17] studied the issue of determining the current state of the system modeled by a nondeterministic finite state automaton and showed a procedure to design an observer for a discrete event process. Ozveren and Willsky [16] defined the observability as having perfect knowledge of the current state at points in time separated by bounded numbers of transitions. An approach is proposed to build an observer according to a finite string from outputs. Besides, they showed that an observer may have an exponential number of states although it can be built in polynomial time.

Shu et al. [19], [20] addressed the detectability, an issue derived from state estimation, for non-probabilistic and probabilistic DES. In [19] four types of detectabilities are defined, and necessary and sufficient conditions are proposed to check these detectabilities. In [20] a given probabilistic automaton is first converted into

⋆ This paper was not presented at any IFAC meeting. Corresponding author Sébastien Lahaye.

Email addresses: aiwen.lai@etud.univ-angers.fr (Aiwen Lai), sebastien.lahaye@univ-angers.fr (Sébastien Lahaye), giua@diee.unica.it (Alessandro Giua).

a non-probabilistic automaton, then necessary and sufficient conditions are proposed to check the strong and weak detectabilities. Keroglou and Hadjicostis [13] investigated the state estimation of probabilistic automata by defining the concept of AA-detectability. As the number of observed output symbols increases, the state estimation will become more accurate. Yin [24] investigated the problem of initial-state detection of probabilistic finite-state automata. An approach is proposed for the verification of stochastic initial-state detectability, and the author proved that this verification problem is PSPACE-complete.

Giua and Seatzu [11] proposed an algorithm for marking estimation, assuming that the PN structure is known and the transition firings can be precisely observed. Labeled PNs with silent or/and indistinguishable transitions are studied in [4], [5], [10]. The notion of *basis marking* is proposed to compute the consistent markings. The authors proved that consistent markings can be characterized by a linear system with a fixed structure that does not change as the length of the observed word increases.

Bonhomme [3] proposed a method for current state estimation in P-time PNs. A state observer is synthesized according to the corresponding untimed PN, and the schedulability of a firing sequence is defined. The checking of schedulability of a sequence is completed by solving a linear programming problem. In [6], timed labeled PNs with unobservable transitions are described by algebraic models. An approach for reconstructing the sequence of unobservable transitions, i.e., completing an observed word into a fireable sequence which is consistent with this observation is developed therein. Timed choice-free PNs are studied in [22]. The authors computed the set of basis markings, notion extended from [5], and used the time equations to reduce this set by removing all markings that are inconsistent with the time information. In [2] an online approach based on the modified state class graph is presented for state estimation of time labeled PNs. The computational complexity of this approach is exponential in the length of the observation.

Besides automata and PNs, max-plus algebra plays an important role in some applications of timed DES [14]. Hardouin et al. [12] investigated the state estimation of timed event graphs (TEGs), a subclass of timed PNs capturing only synchronization and delay phenomena. TEGs can be represented by state-space equations in max-plus algebra, strongly reminiscent of discrete-time representations for conventional linear systems. The design of an observer matrix for TEGs is proposed in analogy to Luenberger observer for classical linear systems.

Max-plus automata are also significant mathematical tools for modeling timed DES involving not only synchronization but also resource sharing. They were investigated as a formalism for DES, first by Gaubert in [7], as a generalization of both conventional automata (by including weights with state transitions) and max-plus linear systems (by capturing resource sharing phenomena). This paper deals with the state estimation problem of a system represented as a max-plus automaton. Max-plus automata represent a particular class of weighted automata and if a timed interpretation is given to weights, then max-plus automata are strongly related to timed automata [1]. There exist connections, and in some cases equivalences, between max-plus automata and time or timed PNs [8],[15], the models used in [2], [3], [6], [22]. However, to the best of our knowledge, there is no work for dealing with this problem when max-plus automata are used as the system models in the literature. According to an observed timed sequence, the state estimation problem consists in finding all possible states in which the system may be at the given time instant.

We first give the definition of the set of states consistent with an observation at a given time. Then we propose algorithms to solve online the state estimation problem. The main idea behind the proposed algorithms originates from the fact that the dynamic behavior of a max-plus automaton can be characterized by its state vector, solution of recurrent equations on words representing the sequence of occurring events.

This paper is structured as follows. In Section 2, we briefly recall some necessary concepts on max-plus algebra, max-plus automata and sequence projection. Section 3 presents the problem statement and specifies our assumptions. In Subsection 4.1, we propose algorithms to estimate the consistent states for any observed timed sequence over the operation of the system. Following the proposed algorithms, one numerical example is shown to illustrate them. In Subsection 4.2, we analyse the computational complexity of state estimation for a fixed-length observation using our approach. Finally, conclusions and future work are drawn in Section 5.

2 Background

2.1 Max-plus algebra

Definition 1 *An idempotent semiring (Dioid) is a set $\mathcal{D}$ together with two binary operations (addition and multiplication), denoted respectively $\oplus$ and $\otimes$. $\mathcal{D}$ with $\oplus$ forms an idempotent semigroup, i.e., $\oplus$ is commutative, associative, has a zero element ε ($\varepsilon \oplus a = a$ for each $a \in \mathcal{D}$), and is idempotent: $a \oplus a = a$ for each $a \in \mathcal{D}$. $\otimes$ is associative, has a unit element e, and distributes over $\oplus$. Moreover, ε is absorbing for $\otimes$, i.e., $\forall a \in \mathcal{D}$, $\varepsilon \otimes a = a \otimes \varepsilon = \varepsilon$.*

Example 1 *The max-plus algebra, denoted by $\mathbb{R}_{max}$, is a typical instance of dioid with $\mathcal{D} = \mathbb{R} \cup \{-\infty\}$. The maximum plays the role of addition $\oplus$, and the conventional addition is used as multiplication $\otimes$, with $\varepsilon = -\infty$, $e = 0$.*

Example 2 *Let E be an alphabet, i.e., a finite non empty set of symbols. A string ω defined on E is a sequence (concatenation) of symbols in E, e.g., $\omega = e_1 e_2 \cdots e_k$ with $e_1, e_2, \cdots, e_k \in E$, and k is called its length. The empty word, the sequence of zero length, is denoted by ϵ. The free monoid E^* defined on E is the set of all strings defined on E. Formal languages are subsets of the free monoid E^*. The set of formal languages with the union of languages playing the role of addition and concatenation of languages playing the role of multiplication is a dioid.*

For matrices $A, B \in \mathcal{D}^{m\times n}$, and $C \in \mathcal{D}^{n\times p}$, the matrix sum and product are defined in a conventional way:

$$[A \oplus B]_{ij} \triangleq A_{ij} \oplus B_{ij} = max(A_{ij}, B_{ij}),$$
$$[A \otimes C]_{ij} \triangleq \bigoplus_{k=1}^{n} (A_{ik} \otimes C_{kj}) = \max_{k=1,\cdots,n} (A_{ik} + C_{kj}).$$

2.2 Max-plus automata

An automaton with weights in the $\mathbb{R}_{max}$ semiring is called a max-plus automaton [7].

Definition 2 *A max-plus automaton is a tuple $G = (Q, E, \alpha, \mu, \beta)$ where*

- *Q and E are respectively a non-empty finite set of states and an alphabet;*
- *$\alpha \in \mathbb{R}_{max}^{1\times|Q|}$ specifies the initial delays. A state $q \in Q$ is said to be an initial state iff $\alpha_q \neq \varepsilon$, and α_q is the corresponding initial delay;*
- *μ: $E \to \mathbb{R}_{max}^{|Q|\times|Q|}$ is a morphism representing the state transitions given by the family of matrices $\mu(a) \in \mathbb{R}_{max}^{|Q|\times|Q|}$, $a \in E$. For any string $\omega = a_1 a_2 \cdots a_n$, we have $\mu(\omega) = \mu(a_1 a_2 \cdots a_n) = \mu(a_1) \otimes \cdots \otimes \mu(a_n)$;*
- *$\beta \in \mathbb{R}_{max}^{|Q|\times 1}$ specifies the final delays. A state q is said to be a final state iff $\beta_q \neq \varepsilon$, and β_q is the corresponding final delay.*

An automaton G can be associated with a useful graphical representation, which is a valued multigraph:

- Q corresponds to the set of nodes;
- an initial state $q \in Q$ (i.e., $\alpha_q \neq \varepsilon$) is characterized by an input arrow labeled by α_q representing its initial delay. Let us denote $Q_i \subseteq Q$ the set of initial states;
- a final state $q \in Q$ (i.e., $\beta_q \neq \varepsilon$) is characterized by an output arrow labeled by β_q representing its final delay;
- there exists an arrow from q to q', labeled by $a/\mu(a)_{qq'}$, iff $\mu(a)_{qq'} \neq \varepsilon$, $a \in E$. It represents the state transition when event a occurs. In this paper, $\mu(a)_{qq'}$ is interpreted as the activation time for event a before it can occur for the transition from q to q'.

Note that, we assume the states of G are well ordered by natural numbers. With an abuse of notation, we denote $\mu(a)_{qq'}$ the element in the q^{th} row and q'^{th} column of the matrix $\mu(a)$. A max-plus automaton may be nondeterministic because there may be more than one element different from ε in α or/and in a row of $\mu(a)$, $a \in E$.

Example 3 *Assume a max-plus automaton with set of states $Q = \{1, 2, 3, 4\}$, alphabet $E = \{u, b\}$, transitions $\mu(u)_{1,2} = 1$, $\mu(u)_{4,3} = 2$, $\mu(b)_{2,2} = 6$, $\mu(b)_{2,1} = 4$, $\mu(b)_{3,2} = 3$, initial and final delays $\alpha_1 = \alpha_4 = \beta_1 = 0$. The other values for α, β and μ are equal to ε. This max-plus automaton is represented in Fig. 1.*

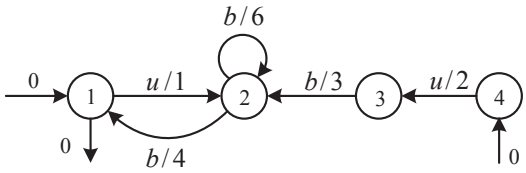

Fig. 1. Max-plus automaton G

Definition 3 *Given a max-plus automaton $G = (Q, E, \alpha, \mu, \beta)$, we define a path of length k as a sequence of transitions $\pi = (q_0, e_1, q_1)(q_1, e_2, q_2) \cdots (q_{k-1}, e_k, q_k)$ where $\mu(e_i)_{q_{i-1}q_i} \neq \varepsilon$, for $i = 1, \cdots, k$.*

Moreover, π is said to be labeled by $e_1 e_2 \cdots e_k$, and π is a circuit if q_0 coincides with q_k. We denote $\mathbb{W}(\pi)$ the product $\otimes$ of the weights on path π as:

$$\mathbb{W}(\pi) = \bigotimes_{i=1,\cdots,k} \mu(e_i)_{q_{i-1}q_i} = \sum_{i=1}^{k} \mu(e_i)_{q_{i-1}q_i}.$$

Let $p, q \in Q$, $\omega \in E^*$. We denote $p \overset{\omega}{\rightsquigarrow} q$ the set of paths from p to q which are labeled by string ω. For $P, R \subseteq Q$, $P \overset{\omega}{\rightsquigarrow} R$ represents the union of $p \overset{\omega}{\rightsquigarrow} q$ for all $p \in P$, $q \in R$. It can be shown that

$$\mu(e_1 e_2 \cdots e_k)_{q_0 q_k} = \bigoplus_{\pi \in q_0 \overset{e_1 \cdots e_k}{\rightsquigarrow} q_k} \mathbb{W}(\pi).$$

This means that $\mu(e_1 e_2 \cdots e_k)_{q_0 q_k}$ represents the maximal weight for paths from q_0 to q_k labeled by $e_1 e_2 \cdots e_k$. The dynamic evolution of the max-plus automaton is described by its state vector $x(\omega) \in \mathbb{R}_{max}^{1\times|Q|}$ which is

defined as:

$$x(\omega) = \begin{cases} \alpha, & if\ \omega = \epsilon; \\ \alpha \otimes \mu(\omega), & otherwise. \end{cases}$$

For any string $\omega = e_1 \cdots e_k \in E^*$, it can be checked that

$$x(\omega)_{q_k} = \max_{q_0 \in Q_i;\ q_1,\cdots,q_k \in Q} \alpha_{q_0} + \left[\sum_{i=1}^{k} \mu(e_i)_{q_{i-1}q_i}\right].$$

In simple words, $x(\omega)_{q_k}$ is the maximal weight of paths labeled by ω going from an initial state to state q_k (the initial delay is added to the weight of the path). We shall interpret $x(\omega)_{q_k}$ as the date at which state q_k is reached when the sequence ω is completed, with the convention that $x(\omega)_{q_k} = \varepsilon$ if q_k is not reachable via ω from an initial state.

Example 4 *Consider the max-plus automaton G in Fig. 1. It can be checked that $x(u) = (\varepsilon, 1, 2, \varepsilon)$, $x(ub) = (5, 7, \varepsilon, \varepsilon)$, and $|Q_i \overset{ub}{\rightsquigarrow} 2| = 2$ since $\pi_1 = (1, u, 2)(2, b, 2)$ and $\pi_2 = (4, u, 3)(3, b, 2)$ are the two paths labeled by ub from an initial state to state 2. When u is completed, state 2 is reached at time 1 because $x(u)_2 = 1$. Besides, $x(ub)_2 = 7$ means that state 2 is reached at instant 7 when ub is completed. Here we observe that events along path π_1 occur as soon as their activation time has elapsed. However, this is not always the case. In particular, along π_2, state 3 is first reached at time 2 since $x(u)_3 = 2$ and state 2 is then reached at instant 7 when ub is completed. Whereas the activation time of b for transition $(3, b, 2)$ is equal to $\mu(b)_{3,2} = 3$, this event here occurs 5 units of time (the real duration) after that state 3 was reached. This is consistent with the interpretation that $\mu(b)_{3,2}$ specifies the activation time before event b can occur for this transition.*

Formally, considering the occurrence of $a \in E$ for transition (q, a, q'), $\mu(a)_{qq'} \neq \varepsilon$, suppose that q is reached at time t_q, it may happen that event a occurs at a time strictly greater than $t_q + \mu(a)_{qq'}$. This can happen if $\exists q \in Q, \exists \omega \in E^* : |Q_i \overset{\omega}{\rightsquigarrow} q| \geq 1$ (i.e., several paths labeled by ω from an initial state to q). Such a configuration captures a synchronization occurring in the system, a common phenomenon in DES.

Based on the above discussion, we can say that a max-plus automaton describes a system characterized by synchronizations between concurrent sequential timed processes. A path in a max-plus automaton represents a concurrent process whose transitions weights denote the minimal time required by the process to fire the transitions. The timed evolution of the automaton will be given by the synchronization of the evolutions of processes corresponding to paths with the same labeling and reaching the same state. Therefore, we associate to each path π of max-plus automaton G a timed sequence corresponding to this synchronization.

Definition 4 *Given a max-plus automaton $G = (Q, E, \alpha, \mu, \beta)$ and a path*

$$\pi = (q_0, e_1, q_1)\,(q_1, e_2, q_2) \cdots (q_{k-1}, e_k, q_k)$$

where $q_0 \in Q_i$, we define the timed sequence $\sigma(\pi) \in (E \times \mathbb{R})^*$ generated by π as: $\sigma(\pi) = (e_1, \tau_1)(e_2, \tau_2) \cdots (e_k, \tau_k)$ where τ_j, $j = 1, \cdots, k$, is defined by: $\tau_j = x(e_1, \cdots, e_j)_{q_j}$.

We use notation $q_0 \overset{\sigma(\pi)}{\rightsquigarrow} q_k$ to denote the fact that path π starts from initial state q_0 and generates the timed sequence $\sigma(\pi)$ reaching state q_k. Such a generated timed sequence consists of pairs composed by an event and a time value. In simple words, it specifies a sequence of events, i.e., the untimed sequence generated by π, and their occurrence time instants. In this paper, for any generated timed sequence σ, we denote $t_f(\sigma)$ the time instant at which the last event in σ occurred, i.e., the completion time of σ. For any $\sigma_1, \sigma_2 \in (E \times \mathbb{R})^*$, $\sigma_1 \cdot \sigma_2$ is used to represent their concatenation.

Example 5 *Consider again path $\pi_1 = (1, u, 2)(2, b, 2)$ and path $\pi_2 = (4, u, 3)(3, b, 2)$ of G in Fig. 1 with $Q_i = \{1, 4\}$. The generated timed sequences are $\sigma(\pi_1) = (u, 1)(b, 7)$ and $\sigma(\pi_2) = (u, 2)(b, 7)$. Therefore, $t_f(\sigma(\pi_1)) = t_f(\sigma(\pi_2)) = 7$.*

Based on the definition of timed sequence generated by a path π, we define the generated timed language of a max-plus automaton.

Definition 5 *Given a max-plus automaton $G = (Q, E, \alpha, \mu, \beta)$, the generated timed language $L(G)$ (set of all timed sequences generated by G) is defined as:*

$$\begin{aligned} L(G) = \{&\sigma \in (E \times \mathbb{R})^* \mid \exists q \in Q, \\ &\exists \omega \in E^*,\ \exists \pi \in Q_i \overset{\omega}{\rightsquigarrow} q :\ \sigma(\pi) = \sigma\} \end{aligned} \tag{1}$$

2.3 Sequence projection

In the state estimation problem, the alphabet is usually partitioned as $E = E_o \cup E_{uo}$ where E_o (resp. E_{uo}) is the set of observable (resp. unobservable) events.

Definition 6 *Given an alphabet $E = E_o \cup E_{uo}$, the projection operator on the set of observable events E_o is denoted by $P : E^* \to E_o^*$ and is defined as: $P(\epsilon) = \epsilon$; $P(sa) = P(s)a$, if $a \in E_o$, otherwise $P(sa) = P(s)$.*

Now we extend the projection operator to any timed sequence $\sigma = (e_1, t_1)(e_2, t_2) \cdots (e_k, t_k) \in (E \times \mathbb{R})^*$. The projection of a timed sequence on observable pairs is

denoted by $P : (E \times \mathbb{R})^* \to (E_o \times \mathbb{R})^*$ where $P(\sigma)$ is the string obtained from σ by erasing all pairs corresponding to unobservable events. For instance, consider $\sigma(\pi) = (a,1)(u,4)(a,5)$ with a observable and u unobservable. Then, we have $P(\sigma(\pi)) = (a,1)(a,5)$. In the rest of this paper, we denote $P(L(G))$ the set of possibly observed timed sequences for max-plus automaton G. In other words, all observations obtained by an external agent belong to $P(L(G))$.

3 Problem statement

In this paper, we deal with the problem of estimating the state of a timed DES represented as a max-plus automaton whose state cannot be directly detected. The studied max-plus automaton is completely known, namely, initial states, state transitions, weights of transitions are available. Considering that the firing of unobservable events cannot be distinguished by an external agent observing the system evolution, we assume that all the unobservable events are labeled by the same symbol u, i.e., $E_{uo} = u$ and $E = E_o \cup \{u\}$. We make the following assumption.

Assumption 1 *There is no circuit labelled only by unobservable events.*

Assumption 1 implies that generated sequences of unobservable events (strings composed exclusively of u) have finite length.

For any state $q \in Q$, we use notation $q^\bullet$ to represent the set of its output transitions, i.e., $q^\bullet = \{(q,a,q') | a \in E, q' \in Q : \mu(a)_{qq'} \neq \varepsilon\}$. Given a timed sequence $\sigma = (e_1,t_1)(e_2,t_2)\cdots(e_k,t_k)$, we denote $lab(\sigma) = e_1e_2\cdots e_k$ the sequence of labels associated with σ, neglecting the occurrence dates.

Given an observed timed sequence $\sigma_o \in P(L(G))$ and a time instant τ, we define the set $C(\sigma_o,\tau)$ of $(\sigma_o,\tau)-$consistent states as the set of all possible states in which the system may be at τ after the observation of σ_o. We assume that there is no further observation after the last event in σ_o.

Definition 7 *Given an observed timed sequence $\sigma_o \in P(L(G))$ and a time instant $\tau \geq t_f(\sigma_o)$, the set of all $(\sigma_o,\tau)-$* consistent states *is defined as*

$$\begin{aligned} C(\sigma_o,\tau) = \{q \in Q \mid (\exists \sigma \in L(G), \exists q_0 \in Q_i : q_0 \overset{\sigma}{\rightsquigarrow} q, \\ P(\sigma) = \sigma_o, t_f(\sigma) \leq \tau) \wedge ((q^\bullet = \emptyset) \vee \\ (\exists (q,a,q') \in q^\bullet : \tau < x(lab(\sigma)a)_{q'}))\} \end{aligned} \tag{2}$$

In simple words, a state q without output transition is a consistent state, if there exists a timed sequence σ from an initial state to q such that the projection of σ coincides with the observation, and the completion time of σ is less than or equal to given time instant τ. If q has output transition(s), then at least one of the output transitions is required to occur after τ.

Problem 1 *The state estimation problem of a max-plus automaton consists in finding a systematic approach to characterize the set $C(\sigma_o,\tau)$ for any observation σ_o and any time instant $\tau \geq t_f(\sigma_o)$.*

Example 6 *Consider again the automaton G in Fig. 1. Let $\sigma_o = (b,7)(b,11)$ and $\tau = 11.5$. It can be verified that the set of consistent states is $C(\sigma_o,\tau) = \{1\}$. In fact, $\pi = (1,u,2)(2,b,2)(2,b,1)$ is the unique path whose timed sequence is consistent with σ_o. We have $\sigma = \sigma(\pi) = (u,1)(b,7)(b,11)$, $1 \overset{\sigma}{\rightsquigarrow} 1$, $t_f(\sigma) = 11 \leq 11.5$, and $P(\sigma) = \sigma_o$. Moreover, the occurrence time instant of the output transition $(1,u,2)$ of state 1 is $x(ubbu)_2 = 12$, which is greater than 11.5. Therefore, state 1 is consistent with σ_o and τ.*

4 State estimation of max-plus automata

4.1 State estimation

In this subsection we introduce an approach to solve Problem 1.

Definition 8 *Given a max-plus automaton $G = (Q,E,\alpha,\mu,\beta)$, a pair (q_j,ω_j) with $q_j \in Q$ and $\omega_j \in E^*$ is said to be* compatible *with a timed sequence $\sigma_o = (e_1,t_1)(e_2,t_2)\cdots(e_j,t_j) \in P(L(G))$, $j \geq 1$, if there exists path π labeled by ω_j ending with e_j from an initial state q_0 in G such that $P(\sigma(\pi)) = \sigma_o$ and $q_0 \overset{\sigma(\pi)}{\rightsquigarrow} q_j$. We define $W(\sigma_o)$ as the set of all pairs compatible with σ_o.*

We define two functions $UpdateW$ and $ConsistentSet$, given in Algorithms 1 and 2 respectively.

Function $UpdateW$ is employed to update the compatible set each time a new event (e,t) is observed.

Proposition 1 *Assume $W(\epsilon) = \{(q,\epsilon) \mid q \in Q_i\}$. Called for successive pairs in sequence $\sigma_o = (e_1,t_1)(e_2,t_2)\cdots(e_j,t_j) \in P(L(G))$, function UpdateW returns $W(\sigma_o)$.*

Proof: Proposition 1 can be proved by induction on the length of the observation. $W(\epsilon) = \{(q,\epsilon) \mid q \in Q_i\}$ is the compatible set for the empty timed sequence. Let $\sigma'_o = (e_1,t_1)(e_2,t_2)\cdots(e_{j-1},t_{j-1})$, and assume that function $UpdateW$ returns $W(\sigma'_o)$ when it has been called for successive pairs in σ'_o. When called for (e_j,t_j), function $UpdateW$ first computes the state vectors $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e_j)\right]_q$ and $\left[\mu(u^i) \otimes \mu(e_j)\right]_{q'q}$, $i = 0,\cdots,|Q|-1$, for each pair $(q',\omega') \in W(\sigma'_o)$

Algorithm 1. Update the compatible set
Input: $W(\sigma_o)$, a pair $(e,t) \in (E_o \times \mathbb{R})$
Output: $W(\sigma_o \cdot (e,t))$
function $UpdateW((e,t), W(\sigma_o))$
  $W(\sigma_o \cdot (e,t)) := \emptyset$
  for each $(q', \omega') \in W(\sigma_o)$ **do**
    for each $q \in Q$ **do**
      for $i = 0$ to $|Q| - 1$ **do**
        if $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e)\right]_q = t$
          and $\left[\mu(u^i) \otimes \mu(e)\right]_{q'q} \neq \varepsilon$ **then**
          $W(\sigma_o \cdot (e,t)) := W(\sigma_o \cdot (e,t)) \cup \left\{(q, \omega' u^i e)\right\}$
        end if
      end for
    end for
  end for
  return $(W(\sigma_o \cdot (e,t)))$
end function

and for any $q \in Q$. Then, any pair $(q, \omega' u^i e_j)$ is included in set $W(\sigma_o)$ if $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e_j)\right]_q = t_j$ and $\left[\mu(u^i) \otimes \mu(e_j)\right]_{q'q} \neq \varepsilon$. Note that the condition $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e_j)\right]_q = t_j$ ensures that state q can be reached at time instant t_j via string $\omega' u^i e_j$ and the condition $\left[\mu(u^i) \otimes \mu(e_j)\right]_{q'q} \neq \varepsilon$ ensures that there exists a path labeled by $u^i e_j$ from state q' to q. Since (q', ω') is compatible, the above two conditions ensure that $(q, \omega' u^i e_j)$ is compatible with σ_o. Therefore, function $UpdateW$ returns $W(\sigma_o)$ when it has been called for successive pairs in σ_o. □

Function $ConsistentSet$ is employed to determine the set of states consistent with an observation σ_o and a time instant $\tau \geq t_f(\sigma_o)$. Although we assume that no observable event occurs between $t_f(\sigma_o)$ and τ, unobservable events may occur between them. This requires us to enumerate unobservable transitions in continuation of compatible pairs in $W(\sigma_o)$.

Proposition 2 *For any $\sigma_o \in P(L(G))$ whose compatible set is $W(\sigma_o)$ calculated by Algorithm 1, and for a time instant $\tau \geq t_f(\sigma_o)$, the set $C(\sigma_o, \tau)$ returned by function ConsistentSet is equal to $C(\sigma_o, \tau)$ defined in Equation (2).*

Proof: According to Algorithm 1, we know that $W(\sigma_o)$ is composed of all pairs (q', ω') such that q' is reached at time $t_f(\sigma_o)$ from an initial state by a path π labeled by ω' ending with the last event in σ_o, and $P(\sigma(\pi)) = \sigma_o$. For each pair $(q', \omega') \in W(\sigma_o)$ and for any $q \in Q$, function $ConsistentSet$ computes the state vectors $\left[x(\omega') \otimes \mu(u^i)\right]_q$ and $\left[\mu(u^i)\right]_{q'q}$, $i = 0, \cdots, |Q| - 1$. Conditions $\varepsilon \neq \left[x(\omega') \otimes \mu(u^i)\right]_q \leq \tau$ and $\left[\mu(u^i)\right]_{q'q} \neq \varepsilon$ ensure that state q can be reached by a generated timed sequence σ such that $P(\sigma) = \sigma_o$ and $t_f(\sigma) \leq \tau$. Besides, the existence of transition $(q, a, q'') \in q^\bullet$ such that $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(a)\right]_{q''} > \tau$ guarantees that at least one of the occurrence time instants of the output transitions of q is greater than τ. Condition $q^\bullet = \emptyset$ ensures that q has no output transition. All states satisfying the above conditions is included in $C(\sigma_o, \tau)$, which coincides with the definition of $(\sigma_o, \tau)-$consistent states. □

Algorithm 2. Compute the set of consistent states
Input: $W(\sigma_o)$, $\tau \geq t_f(\sigma_o)$
Output: $C(\sigma_o, \tau)$
function $ConsistentSet(W(\sigma_o), \tau)$
  $C := \emptyset$
  for each $(q', \omega') \in W(\sigma_o)$
    for $i = 0$ to $|Q| - 1$ **do**
      $Q_\tau := \emptyset$
      for each $q \in Q$ **do**
        if $\varepsilon \neq \left[x(\omega') \otimes \mu(u^i)\right]_q \leq \tau$
          and $\left[\mu(u^i)\right]_{q'q} \neq \varepsilon$ **then**
          $Q_\tau := Q_\tau \cup \{q\}$
        end if
      end for
      for $q \in Q_\tau$
        $T_q := \emptyset$
        if $q^\bullet = \emptyset$ **then**
          $C := C \cup \{q\}$
        else
          for each $(q, a, q'') \in q^\bullet$
            $T_q = T_q \cup \left\{\left[x(\omega') \otimes \mu(u^i) \otimes \mu(a)\right]_{q''}\right\}$
            if $\exists t \in T_q$ such that $t > \tau$ **then**
              $C := C \cup \{q\}$
            end if
          end for
        end if
      end for
    end for
  end for
  return $(C(\sigma_o, \tau) = C)$
end function

Using functions proposed in Algorithms 1 and 2, Algorithm 3 is introduced to deal with the state estimation problem during the operation of the system.

Proposition 3 *Algorithm 3 provides an online solution to Problem 1.*

Proof: Algorithm 3 initializes $W(\epsilon) = \{(q, \epsilon) \mid q \in Q_i\}$ since all initial states can be reached via an empty string (no observation has occurred). Each time a new observable event occurs, function $UpdateW$ is called to update the compatible set, and, when needed, the consistent states is determined by calling function $ConsistentSet$. According to Propositions 1 and 2, Algorithm 3 does provide a solution to Problem 1. □

Example 7 *Let us consider again the max-plus automa-*

Algorithm 3. Online state estimation
Input: $G = (Q, E, \alpha, \mu, \beta)$
Output: $C(\sigma_o, \tau)$
$\sigma_o := \epsilon$, $W(\sigma_o) := \emptyset$
for each $q \in Q_i$ **do**
  $W(\sigma_o) := W(\sigma_o) \cup \{(q, \epsilon)\}$
end for
while state estimation is required **do**
  if observation (e, t) arrives **then**
    $W(\sigma_o \cdot (e,t)) := UpdateW((e,t), W(\sigma_o))$
    $\sigma_o := \sigma_o \cdot (e,t)$
  end if
  if the set of consistent states is required **then**
    Let τ be the current time instant
    $C(\sigma_o, \tau) := ConsistentSet(W(\sigma_o), \tau)$
  end if
end while

ton G in Fig. 1. Let us assume that the timed sequence $\sigma_o = (b,5)(b,10)$ is observed and that the set of consistent states is required at time instant $\tau = 14$. It can be checked that $\alpha = (e, \varepsilon, \varepsilon, e)$, $\beta = (e, \varepsilon, \varepsilon, \varepsilon)$,

$$\mu(u) = \begin{pmatrix} \varepsilon & 1 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 2 & \varepsilon \end{pmatrix}, \quad \mu(b) = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 4 & 6 & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{pmatrix}$$

1) Initialize $W(\epsilon) = \{(1, \epsilon), (4, \epsilon)\}$.

2) Update compatible set when observation $(b, 5)$ arrives:

Firstly, compute $\mu(u^i)$, $i = 0, 1, 2, 3$:

$$\mu(u^0) = \begin{pmatrix} e & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & e \end{pmatrix}, \mu(u^2) = \mu(u^3) = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{pmatrix}$$

Secondly, for pair $(1, \epsilon)$, compute $x(\epsilon) \otimes \mu(u^i) \otimes \mu(b)$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{1,\bullet}$ (the row corresponding to state 1 in matrix $\mu(u^i) \otimes \mu(b)$), $i = 0, 1, 2, 3$: $x(\epsilon) \otimes \mu(u) \otimes \mu(b) = (5, 7, \varepsilon, \varepsilon)$; $x(\epsilon) \otimes \mu(u^i) \otimes \mu(b) = (\varepsilon, \varepsilon, \varepsilon, \varepsilon)$, for $i = 0, 2, 3$; $[\mu(u) \otimes \mu(b)]_{1,\bullet} = (5, 7, \varepsilon, \varepsilon)$; $\left[\mu(u^i) \otimes \mu(b)\right]_{1,\bullet} = (\varepsilon, \varepsilon, \varepsilon, \varepsilon)$, for $i = 0, 2, 3$. According to function UpdateW, any pair $(q, u^i b)$ is recorded as an element of the compatible set if $\left[x(\epsilon) \otimes \mu(u^i) \otimes \mu(b)\right]_q = 5$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{1,q} \neq \varepsilon$. In this case, the only pair $(1, ub)$ is obtained as an element of the updated set W since $[x(\epsilon) \otimes \mu(u) \otimes \mu(b)]_1 = 5$ and $[\mu(u) \otimes \mu(b)]_{1,1} = 5 \neq \varepsilon$.

Thirdly, for pair $(4, \epsilon)$, compute $x(\epsilon) \otimes \mu(u^i) \otimes \mu(b)$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{4,\bullet}$, $i = 0, 1, 2, 3$: $x(\epsilon) \otimes \mu(u) \otimes \mu(b) = (5, 7, \varepsilon, \varepsilon)$; $x(\epsilon) \otimes \mu(u^i) \otimes \mu(b) = (\varepsilon, \varepsilon, \varepsilon, \varepsilon)$, for $i = 0, 2, 3$; $[\mu(u) \otimes \mu(b)]_{4,\bullet} = (\varepsilon, 5, \varepsilon, \varepsilon)$; $\left[\mu(u^i) \otimes \mu(b)\right]_{4,\bullet} = (\varepsilon, \varepsilon, \varepsilon, \varepsilon)$, for $i = 0, 2, 3$.

Similar to the above process, any pair $(q, u^i b)$ is added into the updated set W if $\left[x(\epsilon) \otimes \mu(u^i) \otimes \mu(b)\right]_q = 5$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{4,q} \neq \varepsilon$. In this case, no pair is recorded. Hence, function UpdateW returns $W((b,5)) = \{(1, ub)\}$.

3) Update compatible set when observation $(b, 10)$ arrives: Similar to step 2), any pair $(q, ubu^i b)$ is recorded as an element of the new compatible set if $\left[x(ub) \otimes \mu(u^i) \otimes \mu(b)\right]_q = t_2 = 10$ and $\left[\mu(u^i) \otimes \mu(b)\right]_{1,q} \neq \varepsilon$. In this case, the unique pair $(1, ubub)$ is recorded, and function UpdateW returns $W((b,5)(b,10)) = \{(1, ubub)\}$.

4) Compute set of consistent states required at $\tau = 14$: For pair $(1, ubub)$, compute $x(ubub) \otimes \mu(u^i)$ and $\left[\mu(u^i)\right]_{1,\bullet}$, $i = 0, 1, 2, 3$: $x(ubub) \otimes \mu(u^0) = (10, 12, \varepsilon, \varepsilon)$; $x(ubub) \otimes \mu(u) = (\varepsilon, 11, \varepsilon, \varepsilon)$; $x(ubub) \otimes \mu(u^i) = (\varepsilon, \varepsilon, \varepsilon, \varepsilon)$, for $i = 2, 3$; $\left[\mu(u^0)\right]_{1,\bullet} = (e, \varepsilon, \varepsilon, \varepsilon)$; $[\mu(u)]_{1,\bullet} = (\varepsilon, 1, \varepsilon, \varepsilon)$; $\left[\mu(u^i)\right]_{1,\bullet} = (\varepsilon, \varepsilon, \varepsilon, \varepsilon)$, for $i = 2, 3$.

According to Algorithm 2, we get $Q_\tau = \{1, 2\}$. State 1 is not consistent with $\sigma_o = (b,5)(b,10)$ and $\tau = 14$ since it has only one output transition labeled by u leading to state 2, and $x(ababa)_2 = 11 < \tau$. State 2 is consistent with σ_o and τ because one of its output transitions occur at time instant $x(ababab)_2 = 17$ greater than τ. Therefore $C(\sigma_o, \tau) = \{2\}$.

4.2 COMPUTATIONAL COMPLEXITY

In this subsection, we discuss the computational complexity of estimating all possible states consistent with an observation $\sigma_o = (e_1, t_1)(e_2, t_2) \cdots (e_k, t_k) \in P(L(G))$ and a given time instant $\tau \geq t_f(\sigma_o) = t_k$ using Algorithm 3.

Function *UpdateW* is used to update the compatible set each time an event is observed. Let $\sigma'_o = (e_1, t_1)(e_2, t_2) \cdots (e_{j-1}, t_{j-1})$ and $\sigma''_o = (e_1, t_1)(e_2, t_2) \cdots (e_j, t_j)$ with $1 \leq j \leq k$. In order to calculate $W(\sigma''_o)$ from $W(\sigma'_o)$ when observation (e_j, t_j) arrives, we have to check whether the pair $(q, \omega' u^i e_j)$, $i = 0, \cdots, |Q|-1$, satisfies $\left[x(\omega') \otimes \mu(u^i) \otimes \mu(e_j)\right]_q = t_j$ and $\left[\mu(u^i) \otimes \mu(e_j)\right]_{q'q} \neq \varepsilon$ for each pair (q', ω') belonging to $W(\sigma'_o)$. Therefore, computing $W(\sigma''_o)$ requires $2 \times |Q|^2 \times |W(\sigma'_o)|$ comparisons where $|W(\sigma''_o)|$ is the cardinality of $W(\sigma'_o)$. In the worst case, it is equal to $2 \times |Q|^{j+2}$ since $|W(\sigma'_o)| \leq |Q|^j$. There are

k observed events in σ_o, which means that function *UpdateW* should be called k times. Thus, the total number of comparisons for calculating $W(\sigma_o)$ is at most $\sum_{j=1}^{k} 2 \times |Q|^{j+2}$.

The function *ConsistentSet* is called to calculate all states consistent with observation σ_o and τ. Any state $q \in Q$ is a consistent state if there exists a pair $(q', \omega') \in W(\sigma_o)$ and an integer number $l \in [0, |Q|-1]$ such that $\varepsilon \neq \left[x(\omega') \otimes \mu(u^l)\right]_q \leq \tau$ and $\left[\mu(u^l)\right]_{q'q} \neq \varepsilon$. Besides, the occurrence time instants of output transitions of q need to be compared with τ. In the worst case, the number of comparisons required to check this for all states in the worst case is $(3 + |E|) \times |Q|^{k+3}$.

Therefore, in the worst case, the total complexity of solving the problem of state estimation for a fixed-length observation $\sigma_o = (e_1, t_1)(e_2, t_2) \cdots (e_k, t_k)$ and a given time instant $\tau \geq t_k$ using Algorithm 3 is $\mathcal{O}(\sum_{j=1}^{k} 2 \times |Q|^{j+2} + (3 + |E|) \times |Q|^{k+3})$.

Remark 1 *The computational complexity, in terms of the number of comparisons, of our online algorithm grows exponentially with the length of the observation. This is because each time a new event is observed, the compatible set should be updated from the previous one. For an observation $\sigma_o = (e_1, t_1)(e_2, t_2) \cdots (e_k, t_k)$, all possible sequences of unobservable transitions interleaved with it should be taken into account. That is, at worst, we have to consider all sequences $u^{i_1} e_1 u^{i_2} e_2 \cdots u^{i_k} e_k$ with $i_1, \cdots, i_k = 0, 1, \cdots |Q| - 1$ to determine $W(\sigma_o)$. This implies that in the worst case $|W(\sigma_o)| = |Q|^{k+1}$.*

Note that other approaches for state estimation [2] or related problems [21] for timed DES have also exponential complexity in the length of the observation. A solution to this issue for online implementation could be to construct an observer (as in [18], [19], or similar structures in [2], [4]) to move the computation having high complexity to a preliminary and offline phase. To the best of our knowledge, this problem for max-plus automata is open, and constitutes a challenge for future investigations.

5 Conclusion

Max-plus automata play an important role in the modeling of timed DES, especially in the presence of synchronization. This paper copes with the state estimation problem for a max-plus automaton. Once a timed sequence is observed and a time instant is given, the state estimation problem consists in determining all consistent states, i.e., the states in which the system may be. An algorithm is proposed to deal with this issue online. As future work, we plan to investigate the identification problem, which is closely related to the state estimation problem. In addition, it is planned to study the feasibility of constructing an observer or similar structures for state estimation of max-plus automata.

References

[1] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical computer science*, 126(2):183–235, 1994.

[2] F. Basile, M. Cabasino, and C. Seatzu. State estimation and fault diagnosis of labeled time Petri net systems with unobservable transitions. *IEEE Transactions on Automatic Control*, 60:997–1009, 2015.

[3] P. Bonhomme. Marking estimation of p-time petri nets with unobservable transitions. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45:508–518, 2015.

[4] M. P. Cabasino, A. Giua, M. Pocci, and C. Seatzu. Discrete event diagnosis using labeled Petri nets. An application to manufacturing systems. *Control Engineering Practice*, 19:989–1001, 2011.

[5] D. Corona, A. Giua, and C. Seatzu. Marking estimation of Petri nets with silent transitions. *IEEE Transactions on Automatic Control*, 52:1695–1699, 2007.

[6] P. Declerck and P. Bonhomme. State estimation of timed labeled Petri nets with unobservable transitions. *IEEE Transactions on Automation Science and Engineering*, 11:103–110, 2014.

[7] S. Gaubert. Performance evaluation of (max,+) automata. *IEEE transactions on automatic Control*, 40:2014–2025, 1995.

[8] S. Gaubert and J. Mairesse. Modeling and analysis of timed petri nets using heaps of pieces. *IEEE Transactions on Automatic Control*, 44:683–697, 1999.

[9] A. Giua. State estimation and fault detection using Petri nets. In *Proceedings of 32nd International Conference on Application and Theory of Petri Nets, Newcastle, UK*, pages 38–48, 2011.

[10] A. Giua, D. Corona, and C. Seatzu. State estimation of λ-free labeled Petri nets with contact-free nondeterministic transitions. *Discrete Event Dynamic Systems*, 15:85–108, 2005.

[11] A. Giua and C. Seatzu. Observability of place/transition nets. *IEEE Transactions on Automatic Control*, 47:1424–1437, 2002.

[12] L. Hardouin, C. A. Maia, B. Cottenceau, and M. Lhommeau. Observer design for (max,+) linear systems. *IEEE Transactions on Automatic Control*, 55:538–543, 2010.

[13] C. Keroglou and C. N. Hadjicostis. Verification of detectability in probabilistic finite automata. *Automatica*, 86:192–198, 2017.

[14] J. Komenda, S. Lahaye, J.-L. Boimond, and T. van den Boom. Max-plus algebra in the history of discrete event systems. *Annual Reviews in Control*, 45:240–249, 2018.

[15] S. Lahaye, J. Komenda, and J.-L. Boimond. Compositions of (max,+) automata. *Discrete Event Dynamic Systems*, 25:323–344, 2015.

[16] C. M. Ozveren and A. S. Willsky. Observability of discrete event dynamic systems. *IEEE Transactions on Automatic Control*, 35:797–806, 1990.

[17] P. J. Ramadge. Observability of discrete event systems. In *Proceedings of 25th IEEE Conference on Decision and Control, Athens, Greece*, pages 1108–1112, 1986.

[18] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Diagnosability of discrete-event systems. *IEEE Transactions on Automatic Control*, 40:1555–1575, 1995.

[19] S. Shu, F. Lin, and H. Ying. Detectability of discrete event systems. *IEEE Transactions on Automatic Control*, 52:2356–2359, 2007.

[20] S. Shu, F. Lin, H. Ying, and X. Chen. State estimation and detectability of probabilistic discrete event systems. *Automatica*, 44:3054–3060, 2008.

[21] S. Tripakis. Fault diagnosis for timed automata. In *International symposium on formal techniques in real-time and fault-tolerant systems*, pages 205–221. Springer, 2002.

[22] X. Wang, C. Mahulea, J.e Júlvez, and M. Silva. On state estimation of timed choice-free Petri nets. In *Proceedings of 18th IFAC World Congress, Milano, Italy*, volume 44, pages 8687–8692, 2011.

[23] W. M. Wonham. Towards an abstract internal model principle. *IEEE Transactions on Systems, Man, and Cybernetics*, 6:735–740, 1976.

[24] X. Yin. Initial-state detectability of stochastic discrete-event systems with probabilistic sensor failures. *Automatica*, 80:127–134, 2017.

[25] S. H. Zad, R. H. Kwong, and W. M. Wonham. Fault diagnosis in discrete-event systems: Framework and model reduction. *IEEE Transactions on Automatic Control*, 48:1199–1212, 2003.