Diagnosability of Unambiguous Max-Plus Automata

Aiwen Lai, Jan Komenda, and Sébastien Lahaye

***Abstract*—This article investigates diagnosability and T-diagnosability for discrete-event systems modeled by unambiguous max-plus automata (UMPAs). More precisely, diagnosability requires that the occurrence of any fault can be detected within a finite number of events after the fault has occurred. T-diagnosability requires that the occurrence of any fault can be detected within a delay of at most T time units after its occurrence. First, we propose a polynomial-time algorithm based on the construction of a nondeterministic finite automaton over a weighted alphabet for diagnosability verification of a UMPA. Second, we prove that T-diagnosability of a UMPA can be studied by reducing it to the problem of diagnosability. Third, we introduce an approach to calculate the upper on the time needed for detecting fault occurrence for a diagnosable UMPA, and its complexity is of sixth order in the number of states of the UMPA.**

***Index Terms*—Diagnosability, discrete-event systems (DESs), fault diagnosis, max-plus automata (MPAs).**

I. Introduction

FAULT diagnosis has been extensively studied for automata and Petri nets (PNs) over the past two decades in the framework of discrete-event systems (DESs) [1], [2]. Given a DES, the faults can be associated to some of its events or some of its states. In this work, we assume that certain events of the system are unobservable, and a part of them are faulty. Besides, two different kinds of faults, i.e., intermittent faults and permanent faults, can be considered according to the way of resetting after a fault occurrence. More precisely, if a fault can be reset spontaneously, it is intermittent. In contrast, if a fault can only be reset by a repair performed after the fault is detected, it is permanent. In this work, all faults are considered to be permanent. In other words, once the studied system has reached a faulty state, it cannot return to a normal state.

In [3], the diagnosability is defined for deterministic finite automata (DFAs). A DFA is said to be diagnosable if the failure occurrence can be detected within a delay of at most K event occurrences after the fault occurred. A diagnoser is built to derive a necessary and sufficient condition for checking the diagnosability of a DFA. The number of states of the diagnoser, in the worst case, is exponential with respect to the number of states in the studied DFA. As a result, the checking of diagnosability using a diagnoser is of exponential complexity. In [4]–[6], polynomial-time approaches are proposed to verify the diagnosability of DFAs. In particular, the approach presented in [4] is based on the notion of the verifier, and the algorithm introduced in [5] relies on the construction of a nondeterministic finite automaton (NFA) denoted by G_d therein. The method in [6] constructs a verifier automaton based on the parallel composition of the normal subparts of a DFA with its faulty subpart, which is different from the verifier in [4]. In [7], the modular language diagnosability with local specifications is studied for modular DESs.

In [8]–[10], the integer linear programming technique is used to check the diagnosability for PNs in centralized and decentralized framework. Cabasino *et al.* [11] introduced the notions of *basis reachability diagnoser* and *modified basis reachability graphs* to verify the diagnosability of labeled PNs. In [12], it is proved that the complexity of diagnosability verification in the framework of labeled PNs is EXPSPACE-complete. Mahulea *et al.* [13] studied the effect of fluidization on fault diagnosis for untimed continuous PNs.

Tripakis [14] extended the diagnosability from the logical DESs to the framework of timed automata (TAs), where T-diagnosability is defined. A TA is considered to be T-diagnosable, if any fault occurring in a TA can be detected in T time units after the fault occurs. It is proved that checking of T-diagnosability is PSPACE-complete. Bouyer *et al.* [15] studied the T-diagnosability of two classes of TAs, i.e., deterministic TAs and event-recording TAs. The authors prove that the computational complexity of verifying the existence of a diagnoser for a deterministic TA is 2-EXPTIME-complete, and for an event-recording TA, it is PSPACE-complete.

Zad *et al.* [16] and Pan and Hashtrudi-Zad [17] defined the time-diagnosability for a timed DES, which is modeled by an automaton whose event set contains an event representing the tick of a global clock [18]. An algorithm with polynomial complexity in the number of states of the studied automaton is proposed to verify time-diagnosability. In [19], the diagnosability as well as the T-diagnosability are studied for labeled time PNs. The modified state class graph (MSCG) is first constructed from the studied net. Then, a diagnoser is built from the isomorphism of the MSCG, to verify diagnosability and T-diagnosability.

In [20], a method based on the search for strongly connected components of a diagnoser-like automaton is introduced

Manuscript received December 3, 2019; revised February 11, 2021 and March 11, 2022; accepted May 8, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62103344; in part by the Natural Science Foundation of Fujian Province of China under Grant 2021J05015; in part by the National Key Research and Development Program of China under Grant 2021ZD0112600; in part by Grantová Agentura Ceské Republiky under Grant 19-06175J; and in part by RVO under Grant 67985840. This article was recommended by Associate Editor F. Deng. *(Corresponding author: Aiwen Lai.)*

Aiwen Lai is with the Department of Automation, Xiamen University, Xiamen 361102, China (e-mail: aiwenlai@xmu.edu.cn).

Jan Komenda is with the Institute of Mathematics, Czech Academy of Sciences, 61662 Brno, Czech Republic (e-mail: komenda@math.cas.cz).

Sébastien Lahaye is with the University of Angers, LARIS, SFR MATHSTIC, 49000 Angers, France (e-mail: sebastien.lahaye@univ-angers.fr).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSMC.2022.3176045.

Digital Object Identifier 10.1109/TSMC.2022.3176045

for diagnosability verification of a time-weighted automaton [21]. A procedure is presented to calculate the maximum time to detect the fault occurrences, whose complexity is polynomial in the size of a diagnoser-like automaton. Viana and Basilio [22] extended the above results to decentralized diagnosability (codiagnosability) of timed DESs modeled by weighted automata and introduced a new test for codiagnosability verification based on searching for strongly connected components in a diagnoser-like automaton. The approach in [23] extends the verifier automaton in [6] to K-codiagnosability and T-codiagnosability of timed DESs modeled by weighted automata. Approaches having polynomial time complexity are proposed to calculate the maximum number of event occurrences and the maximum time necessary for detecting the occurrence of faults. However, we emphasize that the above approaches [20], [22], [23] rely on the assumption that the underlying logical automata are (co-)diagnosable. This is restrictive because a timed system can be diagnosable owing to its timing structure in spite of nondiagnosability of the underlying logical system.

In this article, we analyze the diagnosability and T-diagnosability of max-plus automata (MPAs), which is a quantitative extension of NFAs. Unlike traditional automata, each state transition in an MPA carries a weight that belongs to the max-plus semiring. MPAs represent a well-studied class of timed DES models [24], where the weights associated with transitions model the time needed for executing the transitions. In this work, the focus is on a restricted but important class of MPA, i.e., unambiguous MPAs (UMPAs) where no two or more paths leading from an initial state to the same state are labeled by the same string. Note that, unlike [20], [22], and [23], we do not assume that the underlying logical automaton of a UMPA is diagnosable, and use the time information to refine the diagnosability of a timed system.

Let us emphasize that the modeling power of UMPAs is fairly large, because every bounded timed PN under race firing policy can be represented by deterministic MPAs (a subclass of UMPAs) [25], [26]. Moreover, Triska and Moor [26] generalized their results to bounded timed PNs under the so-called open-loop race policy, where solving of conflicts (deciding which transitions are to be fired) is more flexible [27]. The main advantage of the UMPA model is that it is a simple algebraic model meaning we can use purely algebraic techniques to solve problems like T-diagnosability. Note that existing works on fault diagnosis for T-time PNs [19] suffer from severe limitations, e.g., their finite automata abstractions known as state class graphs are not always finite, while we can use exact translations (not abstractions) preserving behaviors in terms of automata with multiplicities in max-plus or interval semirings. Furthermore, existing works on the diagnosability of TAs [14], [15] based on region/zone automata construction suffer from high computational complexity (exponential in many parameters, such as the number of clocks and the granularity of a system). To conclude, we can use the efficient algebraic techniques to decide about diagnosability or to compute T-diagnosability bound, but at the same time, the states of these deterministic representations keep the interpretation in terms of time or timed PNs. Finally, we mention that we are not restricted to a finite horizon as in [23], where the models are based on so-called (time) weighted systems.

We emphasize that time information plays an important role in the diagnosability verification for UMPAs. It will be shown later that the diagnosability of a UMPA can be different from that of the underlying logical automaton due to the influence of the transition weights. This work is the first study of diagnosability and T-diagnosability for MPAs in the literature. We propose an approach to verify the diagnosability and T-diagnosability of an important class of MPAs, namely, UMPAs, where no two or more paths are labeled by a given string leading to the same state from an initial state.

The main contributions of the work are as follows.

1) The notions of diagnosability and T-diagnosability are defined for UMPAs.
2) For a given UMPA G, we first construct an NFA over a weighted alphabet and then derive from the NFA a necessary and sufficient condition that can be used to test the diagnosability of G, with a polynomial-time complexity.
3) We prove that UMPA G is T-diagnosable if and only if it is diagnosable. An algebraic approach is presented to calculate the maximum time required to detect the occurrence of a fault in a diagnosable UMPA G. We show that the complexity of the proposed algebraic approach is of sixth order in the number of states of G.

This article is organized as follows. In Section II, we discuss differences and similarities between this work and some related existing work in the timed framework. Section III reviews some preliminaries on MPAs. Section IV formally defines the diagnosability and T-diagnosability for UMPAs. In Section V, an algorithm with polynomial-time complexity is proposed for checking diagnosability of a UMPA. In Section VI, we give an approach for calculating the maximum time for detecting the fault occurrence in a diagnosable UMPA, and its complexity is of sixth order in the number of states of the system. Finally, Section VII summarizes this work and gives some future research directions.

II. Related Works on Diagnosability

In this section, we illustrate the differences between the work in this article and that from some related references on diagnosability in various timed frameworks/models.

In [19], the number of nodes of the MSCG is exponential with respect to the labeled timed PN structure and the number of tokens in the initial marking. The state-space cardinality of the diagnoser is exponential in the number of nodes of the MSCG. Hence, the complexity of checking diagnosability is worse than exponential in the number of states of the time PNs, whereas a polynomial-time approach is proposed in our work.

In [28], it is proved that the codiagnosability verification problem for TAs [29] under bounded resources is 2EXPTIME-complete. However, we have a polynomial-time approach to verify T-diagnosability for timed systems modeled by UMPAs. While UMPAs can be regarded as a specific class of TAs under some conditions, the use of the UMPAs framework in this article enables computationally more efficient verification of diagnosability.

Zad *et al.* [16] and Pan and Hashtrudi-Zad [17] investigated T-diagnosability of a timed DES proposed by Brandin and Wonham [18], i.e., the model under study is an automaton whose event set contains one event representing a global clock tick. It should be noticed that the timed discrete-event models proposed by Brandin and Wonham are not dense real-time systems, while the UMPAs studied in this article form a class of dense real-time systems.

Theapproaches in [6] and [22] are adopted to check the co-diagnosability for networked DESs with timing structure (NDESWTS), where there exist delays and losses of event observations between the measurement sites and local diagnosers [30]. A method using verifiers, which are built from equivalent untimed automata G_{a_i}, is proposed for co-diagnosability of an NDESWTS. Hence, the computational complexity is polynomial in the size of G_{a_i}. Note that the complexity related to the size of the original system NDESWTS is not given in [30] since the upper bound on the number of states and transitions of the untimed automata is not captured therein.

Here, we emphasize the main differences between this work and the aforementioned works [22], [23] on diagnosability and T-diagnosability for weighted automata. First, in [22] and [23], the language generated by a weighted automaton is equal to the language generated by its corresponding logical (i.e., untimed) automaton. This means that the timing of events is an auxiliary structure, and the time information is not used to refine the diagnosability of a timed system. This is not the case in the framework of UMPAs, where the language generated by a UMPA is a set of timed sequences of (event, time-instant) pairs, rather than a set of logical sequences of events. Second, in [22] and [23], there exists the assumption that the corresponding logical automaton of a weighted automaton is (co-)diagnosable, which is not required in this article. In fact, we are concerned with the diagnosability of a UMPA with respect to its timed behavior, which often makes the system diagnosable as a timed system while the underlying logical automaton is not diagnosable.

Finally, we point out the main difference between this work and the work in [31], where the notion of G_v is first presented to check the detectability and opacity of a UMPA. The main differences are the following three aspects. First, in [31], G_v is customized for a UMPA G having a restrictive structure. More precisely, it is required that G has a unique initial state, and a fault can only occur at the beginning of an evolution and can only occur once. There are no such restrictions in this article, and as a result, the construction of G_v is modified. Second, in this work, Lemmas 2 and 3 are introduced to illustrate the relationship between G and G_v, which are not presented in [31]. Finally, if the studied system G is diagnosable, the bounds of K and T, i.e., the maximum number of event occurrences and the maximum time required for detecting the occurrence of a fault, are specified in Remark 3 and Section VI, respectively, none of them are introduced in [31].

III. Preliminaries

In this section, we introduce some preliminaries of max-plus algebra and MPAs [24], [32], [33].

Definition 1: A Dioid is a tuple $(\mathcal{D}, \oplus, \otimes)$, where $\mathcal{D}$ is a set, $\oplus$ and $\otimes$ are two binary operations satisfying the following conditions: $(\mathcal{D}, \oplus)$ forms a commutative monoid with zero element ε, and $\oplus$ is idempotent. $(\mathcal{D}, \otimes)$ forms a monoid. Operation $\otimes$ distributes over $\oplus$, and ε is absorbing for $\otimes$.

The max-plus semiring $\mathbb{R}_{\max}$ is an instance of dioid with $\mathcal{D} = \mathbb{R} \cup \{-\infty\}$, $\oplus = \max$, $\otimes = +$, $\varepsilon = -\infty$ and 0 is the unit element. Let $\mathbb{R}_{\max}^{m\times n}$ be the set of all $m \times n$ matrices, where all elements belong to $\mathbb{R}_{\max}$. Let $A, B \in \mathbb{R}_{\max}^{m\times n}$ and $C \in \mathbb{R}_{\max}^{n\times p}$ be max-plus matrices, then the sum and product are defined as follows:

$$[A \oplus B]_{ij} = A_{ij} \oplus B_{ij} = \max\big(A_{ij}, B_{ij}\big);$$

$$[A \otimes C]_{ij} = \bigoplus_{k=1}^{n} \big(A_{ik} \otimes C_{kj}\big) = \max_{k=1,\ldots,n} \big(A_{ik} + C_{kj}\big).$$

We use E, which consists of a finite number of symbols, to represent an alphabet. A string on E is defined as a sequence of symbols that belong to E. We use E^* to represent the set of all finite strings that are defined over E, including the empty string denoted by ϵ. Besides, with a slight abuse of notations, we use $e \in s$ to denote $e \in E$ is an event in sequence $s \in E^*$. MPAs are timed extensions of finite state automata, where each state transition has a weight that belongs to $\mathbb{R}_{\max}$ semiring [24].

Definition 2: An MPA is defined as a tuple $G = (Q, E, t, \varrho, Q_i)$, where:

1) Q is a finite set of states and E is an alphabet;
2) $t : Q \times E \times Q \to \mathbb{R}_{\max}$ is the transition function that associates to each state transition a value belonging to $\mathbb{R}_{\max}$. $t(p, e, q) \neq \varepsilon$ means that there is a transition from state p to state q labeled by symbol e, and the transition weight is equal to $t(p, e, q)$. By convention, $t(p, e, q) = \varepsilon$ if there is no transition from p to q labeled by e;
3) $\varrho : Q \to \mathbb{R}_{\max}$ is the function specifying the initial delays. $\varrho(q) \neq \varepsilon$ (resp., $\varrho(q) = \varepsilon$) represents that state q is (resp., is not) an initial state, and $\varrho(q) \neq \varepsilon$ is the initial weight of q;
4) Q_i is the set of initial states: $Q_i = \{p \in Q | \varrho(q) \neq \varepsilon\}$.

A path of length k in system G is defined as a sequence of state transitions $\pi = (q_1, e_1, q_2)(q_2, e_2, q_3)\cdots(q_k, e_k, q_{k+1})$, where $q_i \in Q$, $i = 1, \ldots, k+1$, $e_j \in E$ and $t(q_j, e_j, q_{j+1}) \neq \varepsilon$, for $j = 1, 2, \ldots, k$. Path π is labeled by string $e_1 e_2 \cdots e_k \in E^*$. It becomes a circuit if the first state q_1 and last state q_{k+1} are identical. We use $q_1 \overset{\omega}{\rightsquigarrow} q_2$ to represent the set of all paths leading from state q_1 to state q_2 that are labeled by ω. For any $Q_1, Q_2 \subseteq Q$, we denote by $Q_1 \overset{\omega}{\rightsquigarrow} Q_2$ the union of $q_1 \overset{\omega}{\rightsquigarrow} q_2$ for any $q_1 \in Q_1$, $q_2 \in Q_2$.

The following is the definition of unambiguity property of an MPA, which is assumed to hold in this work.

Definition 3: An MPA G is unambiguous if for all $p \in Q$ and for all $\omega \in E^*$, $|Q_i \overset{\omega}{\rightsquigarrow} \{p\}| \leq 1$.

In simple words, an MPA G is unambiguous if for any state p and for any string ω, the number of paths labeled by ω from an initial state to state p is less than or equal to 1. Unambiguity is a property of the underlying logical automaton and it is known that it can be checked in polynomial time.

Remark 1: An MPA G is deterministic if the following conditions hold: 1) G has a single initial state and 2) for any

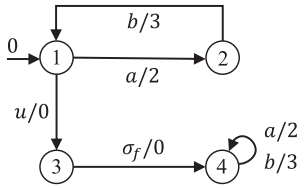

Fig. 1. Unambiguous max-plus automaton G.

state in G, all its output transitions are labeled with different symbols. From this, we know that if G is deterministic, it is unambiguous, but not vice versa. This means that UMPAs represent a more general class of automata than deterministic MPAs.

Example 1: Consider the MPA G in Fig. 1. In this graphical representation of G, the nodes correspond to states of G, the label 0 over the input arrow to state 1 corresponds to the initial weight for this initial state. As an example, the arc from state 1 to state 2 is labeled by $a/2$ since $t(1, a, 2) = 2$. It can be checked that G is deterministic, and hence unambiguous. If now we replace transition $1 \xrightarrow{u/0} 3$ by $1 \xrightarrow{a/3} 3$, then the modified MPA is still unambiguous, but it becomes nondeterministic.

Definition 4: Given a UMPA G and a path $\pi = (q_0, e_1, q_1)(q_1, e_2, q_2)\cdots(q_{n-1}, e_n, q_n)$ where $q_0 \in Q_i$, the timed sequence generated by π is denoted by $\sigma(\pi) \in (E\times\mathbb{R})^*$, and is defined as $\sigma(\pi) = (e_1, \tau_1)(e_2, \tau_2)\cdots(e_n, \tau_n)$, where $\tau_1 = \varrho(q_0) + t(q_0, e_1, q_1)$, $\tau_k = \tau_{k-1} + t(q_{k-1}, e_k, q_k)$ for $k = 2, \ldots, n$.

Intuitively, $\sigma(\pi)$ is a sequence of (event, time-instant) pairs, where the second element in each pair is the occurrence time-instant of the event in the first element. With a slight abuse of notations, we use ϵ_t to denote the empty timed sequence. The concatenation of two timed sequences $\sigma_1, \sigma_2 \in (E \times \mathbb{R})^*$ is a new timed sequence $\sigma = \sigma_1 \cdot \sigma_2$ composed by the sequence of pairs in σ_1 followed by the sequence of pairs in σ_2. The symbol $\cdot$ used to denote concatenation is usually omitted, and one writes $\sigma_1\sigma_2$ instead of $\sigma_1 \cdot \sigma_2$. In particular, for any $\sigma \in (E \times \mathbb{R})^*$, we have $\sigma\epsilon_t = \epsilon_t\sigma = \sigma$.

Definition 5: Given a UMPA G, the timed language generated by G is defined as follows:

$$L(G) = \Big\{x \in (E \times \mathbb{R})^* | \exists q \in Q \quad \exists\omega \in E^*,\ \exists\pi \in Q_i \overset{\omega}{\rightsquigarrow} \{q\} : \sigma(\pi) = x\Big\}. \quad (1)$$

The set of observable events is denoted by $E_o \subseteq E$. The set of unobservable events is denoted by $E_{uo} = E\backslash E_o$.

Definition 6: Given an alphabet $E = E_o \cup E_{uo}$, we use $P : E^* \to E_o^*$ to represent the natural projection to observable sequences, which is defined in the following way:

$$P(\epsilon) = \epsilon$$
$$P(e) = \begin{cases} e, & \text{if } e \in E_o \\ \epsilon, & \text{if } e \in E_{uo} \end{cases}$$
$$P(\omega e) = P(\omega)P(e) \text{ for } \omega \in E^*, e \in E.$$

Natural projection is extended to timed sequences $P : (E \times \mathbb{R})^* \to (E_o \times \mathbb{R})^*$, which is defined as follows:

$$P(\epsilon_t) = \epsilon_t$$
$$P((e, t)) = \begin{cases} (e, t), & \text{if } e \in E_o \\ \epsilon_t, & \text{if } e \in E_{uo} \end{cases}$$
$$P(\sigma(e, t)) = P(\sigma)P((e, t)) \text{ for } \sigma \in (E \times \mathbb{R})^*, (e, t) \in E \times \mathbb{R}.$$

From here we know that for any timed sequence $\sigma = (e_1, t_1)(e_2, t_2)\cdots(e_k, t_k) \in (E \times \mathbb{R})^*$, $P(\sigma)$ is defined as the timed sequence obtained by deleting all the pairs in σ that contain an unobservable event. That is, the unobservable events as well as their corresponding occurrence time instants are removed. We use $P(L(G))$ to represent the projection of the timed language of MPA G, i.e., $P(L(G)) = \{P(\sigma)|\sigma \in L(G)\}$.

IV. Definitions of Diagnosability and T-Diagnosability

Let $E_f = \{\sigma_f\} \subseteq E_{uo}$ denote the set of fault events of MPA (Q, E, t, ϱ, Q_i). The diagnosability problem in this work is to verify if the occurrence of fault event σ_f can be detected after the occurrence of a finite number of observable events or within a finite period of time.

Without loss of generality, we assume that the weights of all initial states of MPA (Q, E, t, ϱ, Q_i) are equal to 0. Note that an MPA with some weights of initial states different from 0 can always be converted to an MPA with all initial weights equal to 0 as follows: for each initial state $q \in Q_i$ with initial weight $\varrho(q) \neq 0$, we add a new state q' to be an initial state with $\varrho(q') = 0$, and add a state transition labeled by empty string ϵ leading from q' to q, with a weight equal to $\varrho(q)$. The following assumptions are needed in this work.

A1: Automaton G is deadlock free, i.e., the number of output transitions of any state in G is greater than 0.

A2: There are no circuits that are only labeled by unobservable events.

A3: Automaton G is unambiguous.

Assumption A1 implies that the studied MPA can evolve indefinitely, this further means that a timed sequence generated by the system can have an infinite length. Assumption A2 states that the length of any timed sequence consisting of only unobservable events is always finite. So far we have not found an approach to address the diagnosability and T-diagnosability verification problem for general MPAs, but this problem becomes solvable with Assumption A3.

For any timed sequence $\sigma = (e_1, t_1)(e_2, t_2)\cdots(e_k, t_k) \in (E \times \mathbb{R})^*$, we denote by $lab(\sigma) = e_1e_2\cdots e_k \in E^*$ the logical sequence of events by ignoring the time instants in σ, by $lab_l(\sigma) = e_k$ the last event in σ, and by $t_l(\sigma) = t_k$ the time instant associated with $lab_l(\sigma)$, i.e., the completion time of σ.

Definition 7 (Diagnosability): A UMPA G is diagnosable with respect to projection $P : E^* \to E_o^*$ and set of fault events $E_f \subseteq E_{uo}$ if the following holds:

$$(\exists K \in \mathbb{N})\big(\forall s \in L(G), lab_l(s) \in E_f\big)$$
$$(\forall v = st \in L(G), |t| \geq K)$$
$$\Rightarrow (\forall\omega \in L(G), P(\omega) = P(v))\big(\exists e_f \in E_f, e_f \in lab(\omega)\big).$$

In simple words, a system is said to be diagnosable if there exists an integer $K \in \mathbb{N}$ such that we can conclude with certainty that a fault belonging to E_f must have happened after at most K event occurrences following the occurrence of a fault.

It should be noticed that the diagnosability of a UMPA can be different from that of the underlying logical automaton due to the influence of the transition weights. For instance, consider the UMPA G in Fig. 5, where a and b are observable normal events, f is an unobservable fault event, and u is an unobservable but normal event. It can easily be checked that the underlying logical automaton is not diagnosable. In fact, there exist two infinite sequences, i.e., $(ab)^k$ and $uf(ab)^k$ with k being arbitrarily large, such that they have the same projection, i.e., $(ab)^k$, one contains fault f, and the other one is fault-free. Therefore, an external agent is never able to determine the occurrence of fault f. However, if we take into account the time information, it will be shown in Example 5 that G is diagnosable.

We make an additional assumption when we deal with T-diagnosability for a UMPA G.

A4: There is no circuit with a sum of transition weights equal to zero in G.

The above assumption means that a system cannot evolve indefinitely without time progress.

Definition 8 (T-Diagnosability): A UMPA $G = (Q, E, t, \varrho, Q_i)$ is T-diagnosable with respect to projection $P : E^* \to E_o^*$ and set of fault events $E_f \subseteq E_{uo}$ if the following holds:

$$(\exists T \in \mathbb{R})\big(\forall s \in L(G), lab_l(s) \in E_f\big)$$
$$(\forall v = st \in L(G), t_l(st) - t_l(s) \geq T)$$
$$\Rightarrow (\forall \omega \in L(G), P(\omega) = P(v))\big(\exists e_f \in E_f, e_f \in lab(\omega)\big).$$

In simple words, a system is said to be T-diagnosable if, after at most T time units following the occurrence of a fault, we can conclude that a fault that belongs to E_f must have happened.

V. Polynomial-Time Algorithm for Diagnosability

In this section, an approach having polynomial-time complexity is presented to verify diagnosability for timed DESs modeled by UMPAs. The presented approach is adapted from the method in [5] dealing with the diagnosability verification of untimed DESs modeled by logical finite state automata.

We transform UMPA $G = (Q, E, t, \varrho, Q_i)$ to its augmented version $G^A = (Q^A, E, t^A, Q_i^A, \varrho^A)$, with respect to the set of faults E_f, where $Q^A \subseteq Q \times \{N, F\}$ is the set of states, $t^A : Q^A \times E \times Q^A \to \mathbb{R}_{\max}$ is the state transition function, $Q_i^A \subseteq Q^A$ is the set of initial states, and $\varrho^A : Q_i^A \to \mathbb{R}_{\max}$ is the function specifying the weights of the initial states. We emphasize that every state in G^A is a pair (q, f), where the first element $q \in Q$ is a state in G and the second element $f \in \{N, F\}$ is a label, specifying the status of state q. More precisely, $f = N$ implies that q is a normal state, and $f = F$ indicates that q is a faulty state. In addition, the number of states in G^A is bounded by $|Q^A| = 2 \times |Q|$.

Algorithm 1 Construction of the Augmented Automaton

Input: A UMPA $G = (Q, E, t, Q_i, \varrho)$.

Output: An augmented automaton $G^A = \big(Q^A, E, t^A, Q_i^A, \varrho^A\big)$ of G.

1: Let $Q_i^A = \emptyset$
2: **for** all $q \in Q_i$ **do**
3: $\quad Q_i^A = Q_i^A \cup \{(q, N)\}$, $\varrho^A((q, N)) = \varrho(q)$
4: **end for**
5: Let $Q^A = Q_i^A$, and assign no tag to states in Q^A.
6: **while** states with no tag exist **do**
7: $\quad$ select a state (q, f) from Q^A
8: $\quad$ **for** all $e \in E$ **do**
9: $\quad\quad$ **if** $e \in E_f$ **then**
10: $\quad\quad\quad R^A((q, f), e) = \big\{(q', F) | q' \in R(q, e)\big\}$
11: $\quad\quad$ **else**
12: $\quad\quad\quad R^A((q, f), e) = \big\{(q', f) | q' \in R(q, e)\big\}$
13: $\quad\quad$ **end if**
14: $\quad\quad$ **for** all $(q', f') \in R^A((q, f), e)$ **do**
15: $\quad\quad\quad$ **if** $(q', f') \notin Q^A$ **then**
16: $\quad\quad\quad\quad Q^A = Q^A \cup \big\{(q', f')\big\}$
17: $\quad\quad\quad\quad$ add no tag to state (q', f').
18: $\quad\quad\quad$ **end if**
19: $\quad\quad\quad t^A((q, f), e, (q', f')) = t(q, e, q')$
20: $\quad\quad$ **end for**
21: $\quad$ **end for**
22: $\quad$ Tag state (q, f) "old".
23: **end while**

For any state $q \in Q$ in UMPA G and for any event $e \in E$, we use $R(q, e)$ to represent the set of states in G that are reachable from q by the occurrence of e, that is, $R(q, e) = \{q' \in Q | t(q, e, q') \neq \varepsilon\}$. Similarly, we denote by $R^A((q, f), e)$ the set of states that are reachable from state (q, f) by the occurrence of event e in G^A, defined as

$$R^A((q, f), e) = \begin{cases} \big\{(q', f) | q' \in R(q, e)\big\}, & \text{if } e \notin E_f \\ \big\{(q', F) | q' \in R(q, e)\big\}, & \text{if } e \in E_f. \end{cases}$$

Algorithm 1 details the construction process of augmented version G^A of UMPA G with respect to E_f.

Steps 1–4 of Algorithm 1 determine the initial states of G^A and the weights of these initial states. The set of initial states Q_i^A of G^A contains all the pairs (q_i, N) such that q_i is an initial state of G. All the second elements in all pairs in Q_i^A are N, which means that no fault events occur when the system starts to run. For all states (q, f) in Q^A with no tag, and for each event $a \in E$, we search for all states (q', f') that are reachable from (q, f) by transitions labeled by a. Note that the weight of transition from state (q, f) to state (q', f') labeled by event a is defined as $t(q, a, q')$. If state (q', f') is not included in Q^A, then we consider it as a new state, and assign no tag to it. We repeat this procedure until there is no unchecked state in Q^A, i.e., all states in Q^A have tag "old."

Example 2: Consider UMPA G in Fig. 1, where $E_o = \{a, b\}$, $E_{uo} = E_f \cup E_{\text{reg}}$ with $E_f = \{\sigma_f\}$ and $E_{\text{reg}} = \{u\}$. By Algorithm 1, the augmented version G^A of G is obtained in Fig. 2.

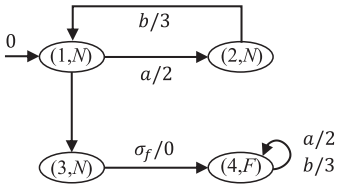

Fig. 2. Augmented automaton G^A of G in Fig. 1.

The following lemma shows the equivalence of the language generated by augmented automaton G^A and the language generated by UMPA G.

Lemma 1: UMPA G and its augmented version G^A have the same generated timed language, that is, $L(G) = L(G^A)$.

Proof: According to the construction process of G^A presented in Algorithm 1, we know that each state in G^A is an extension of the corresponding state in G (i.e., a state q in G corresponds to (q, F) or (q, N) in G^A). A path

$$\pi = ((q_1, f_1), e_1, (q_2, f_2))((q_2, f_2), e_2, (q_3, f_3)) \cdots (q_{k-1}, f_{k-1}), e_{k-1}, (q_k, f_k))$$

belongs to G^A if and only if there exists path $\pi' = (q_1, e_1, q_2)(q_2, e_2, q_3) \cdots (q_{k-1}, e_{k-1}, q_k)$ in G, where $t((q_i, f_i), e_i, (q_{i+1}, f_{i+1})) = t(q_i, e_i, q_{i+1})$ for $i = 1, 2, \ldots k-1$. This implies that G^A is unambiguous, since G is unambiguous. Therefore, by Definition 4 on the timed sequence generated by a path of a UMPA, we have $\sigma(\pi) = \sigma(\pi')$, which further implies that $L(G) = L(G^A)$. ■

We now present a procedure for building an NFA denoted by $G_v = (Q_v, E_v, \delta_v, Q_i^v)$ to verify the diagnosability of a UMPA G, based on the augmented automaton G^A of G. Algorithm 2 details the construction of G_v for a UMPA G.

Let us briefly explain how Algorithm 2 operates. Step 1 constructs augmented automaton G^A of a UMPA G by using Algorithm 1. Step 2 builds a weighted automaton G^o from augmented automaton G^A. The automaton G^o can be seen as nondeterministic projection of automaton G^A. The set of initial states of G^o is initialized as the set of initial states of G^A. For all states q^o in Q^o that have not yet been considered, i.e., states with no tag, and for each observable event $a \in E_o$, we search for all the states q^A that are reachable from q^o by paths labeled by va, strings starting with unobservable sequence of events and ending with event a as the only observable event. If state q^A is not included in Q^o, then we consider it as new, i.e., no tag will be associated to it. We also record the corresponding state transition, i.e., $t^o(q^o, a, q^A) = t(q^o, va, q^A)$. This procedure is repeated until there is no unchecked state in automaton G^o. Step 3 transfers G^o to an NFA $G^o_{\log}$ over weighted alphabet E_o^{log}. Intuitively, in the graphical representation, the labels a/t over the state transitions in G^o are simply replaced by (a, t) in $G^o_{\log}$, which are now considered as new symbols in the alphabet E_o^{log} of $G^o_{\log}$. Step 4 constructs NFA G_v of UMPA G by the parallel composition of $G^o_{\log}$ and $G^o_{\log}$.

Example 3: Consider the augmented automaton G^A in Fig. 2. After applying Algorithm 2, we obtain automata G^o and $G^o_{\log}$ shown on the left and right in Fig. 3, respectively, and obtain G_v shown in Fig. 4.

Algorithm 2 Construction of NFA G_v

Input: UMPA $G = (Q, E, t, Q_i, \varrho)$.

Output: An NFA $G_v = (Q_v, E_v, \delta_v, Q_i^v)$ obtained from G.

1: Compute augmented automaton $G^A = \left(Q^A, E, t^A, Q_i^A, \varrho^A\right)$ of G by Algorithm 1.

2: Build a weighted automaton $G^o = (Q^o, E_o, t^o, Q_i^o, \varrho^o)$ with alphabet E_o from $G^A = \left(Q^A, E, t^A, Q_i^A, \varrho^A\right)$ as follows.
Let $Q^o = Q_i^o = Q_i^A$, $\varrho^o(q) = \varrho^A(q)$ for all $q \in Q_i^o$, and assign no tag to states in Q^o.
while states with no tag exist **do**
select a state $q^o \in Q^o$ with no tag.
for each $a \in E_o$ **do**
for all paths labeled by va with $v \in E_{uo}^*$ leading from state q^o to another state $q^A \in Q^A$ in G^A **do**
if $q^A \notin Q^o$ **then** $Q^o = Q^o \cup \left\{q^A\right\}$, and assign no tag to q^A.
else $t^o(q^o, a, q^A) = t^A(q^o, va, q^A)$.
end if
end for
end for
Tag node q^o "old".
end while

3: Consider weighted automaton $G^o = (Q^o, E_o, t^o, Q_i^o, \varrho^o)$ as a logical NFA $G^o_{log} = (Q^o, E_o^{log}, \delta_o, Q_i^o)$ over a weighted alphabet E_o^{log}, where $E_o^{log} = \left\{(a, \tau) | \exists a \in E_o, \exists q, q' \in Q^o : t^o(q, a, q') = \tau\right\}$, and $\delta_o \subseteq Q^o \times E_o^{log} \times Q^o$ is the transition function: $(q, (a, \tau), q') \in \delta_o$ iff $t^o(q, a, q') = \tau$. Note that $\tau \in \mathbb{R}_{max} \backslash \{-\infty\}$.

4: Compute automaton $G_v = G^o_{log} \parallel G^o_{log}$, the parallel composition of G^o_{log} with itself, as follows:

- $Q_i^v = \left\{(q_1, q_2) | q_1, q_2 \in Q_i^o\right\}$;
- $Q_v' = \left\{(q_1^o, q_2^o) | q_1^o, q_2^o \in Q^o\right\}$;
- $E_v = E_o^{log}$;
- $\delta_v \subseteq Q_v' \times E_v \times Q_v'$ is the set of state transitions. $((p_v^1, p_v^2), (a, \tau), (q_v^1, q_v^2)) \in \delta_v$ iff $(p_v^1, (a, \tau), q_v^1) \in \delta_o$ and $(p_v^2, (a, \tau), q_v^2) \in \delta_o$.
- Let $G_v = (Q_v, E_v, \delta_v, Q_i^v) = Ac(Q_v', E_v, \delta_v, Q_i^v)$.

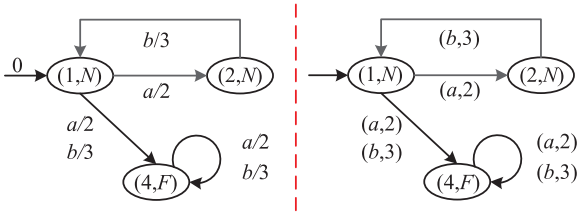

Fig. 3. Automata G^o and $G^o_{\log}$ of G^A in Fig. 2.

Note that the generated language $L(G_v)$ of the NFA G_v is a subset of E_v^*, i.e., $L(G_v) \subseteq E_v^*$, and the generated language $L(G)$ of G is a subset of $(E \times \mathbb{R})^*$, i.e., $L(G) \subseteq (E \times \mathbb{R})^*$. For any observation $\sigma_o = (a_1, \tau_1)(a_2, \tau_2) \cdots (a_n, \tau_n) \in P(L(G))$, we define $\sigma_o^{\text{elem}} = (a_1, \tau_1)(a_2, \tau_2 - \tau_1) \cdots (a_n, \tau_n - \tau_{n-1})$ to represent the equivalent expression of σ_o in E_v^*, where

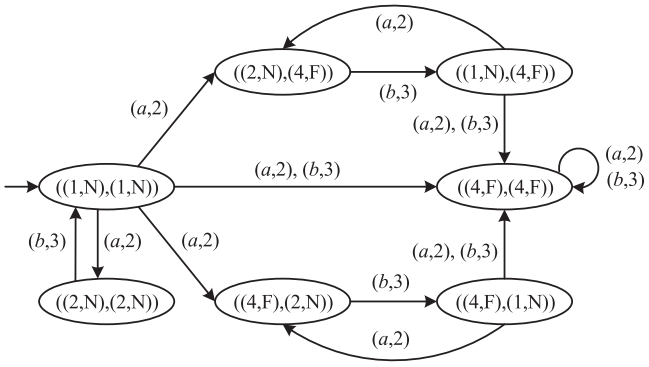

Fig. 4. Diagram of G_v of G in Fig. 1.

$\tau_k - \tau_{k-1}$, $k = 2, 3, \ldots, n$, represents the time for the elementary transition corresponding to observable event a_k. We denote by $P(L(G))^{\text{elem}}$ the equivalent notation of $P(L(G))$, i.e.,

$$P(L(G))^{\text{elem}} = \left\{\sigma_o^{\text{elem}} | \sigma_o \in P(L(G))\right\}.$$

The following two lemmas illustrate some properties of NFA $G^o_{\log}$ and NFA G_v constructed by Algorithm 2.

Lemma 2: NFA $G^o_{\log} = (Q^o, E_o^{log}, \delta_o, Q_i^o)$ and automaton G have the following relationship:

1) $P(L(G))^{\text{elem}} = L(G^o_{\log})$;
2) consider an elementary circuit in $G^o_{\log}$

$$\pi = \left(\left(q_o^1, f_1\right), e_1, \left(q_o^2, f_2\right)\right)\left(\left(q_o^2, f_2\right), e_2, \left(q_o^3, f_3\right)\right) \cdots \left(\left(q_o^n, f_n\right), e_n, \left(q_o^1, f_1\right)\right)$$

with $q_o^k \in Q^o$, $e_k \in E_o^{log}$ and $f_k \in \{N, F\}$ for $k = 1, 2, \ldots, n$. Let $s = e_1 e_2 \cdots e_n$, and assume that q_o^1 is reached by $\omega \in L(G^o_{\log})$ from an initial state in $G^o_{\log}$, then we have:

a) $f_1 = f_2 = \cdots = f_n = N$ or $f_1 = f_2 = \cdots = f_n = F$;
b) if $f_1 = f_2 = \cdots = f_n = N$, then for all $j \in \{1, 2, \ldots, \}$ there exists $\gamma \in L(G)$ such that $P(\gamma)^{\text{elem}} = \omega s^j$, and γ is fault-free, i.e., for any $e_f \in E_f$, $e_f \notin lab(\gamma)$; if $f_1 = f_2 = \cdots = f_n = F$, then for all $j \in \{1, 2, \ldots, \}$ there exists $\gamma \in L(G)$ such that $P(\gamma)^{\text{elem}} = \omega s^j$, and γ contains a fault, i.e., $\exists e_f \in E_f$, $e_f \in lab(\gamma)$.

Proof: 1) From step 2 of Algorithm 2, we know that the state set Q^o of automaton G^o is composed only of the initial states of augmented automaton G^A and of those states in G^A that have at least one input transition labeled by an observable event. Besides, the duration of (a sequence of) unobservable events along a path is added to the next observable event. This means $L(G^o) = P(L(G^A))$. Further, according to step 3 of Algorithm 2, we know that the only difference between NFA $G^o_{\log}$ and automaton G^o is that a label a/t in G^o is replaced by (a, t) in $G^o_{\log}$, which implies $L(G^o_{\log}) = L(G^o)^{\text{elem}}$, where $L(G^o)^{\text{elem}} = \{\sigma^{\text{elem}} | \sigma \in L(G^o)\}$. Therefore, by combining $L(G^A) = L(G)$ in Lemma 1, we have $P(L(G))^{\text{elem}} = L(G^o_{\log})$.

2)-a): The labels (i.e., the second elements) of all states of a circuit in $L(G^o_{\log})$ should be the same, either N or F, since the system cannot recover from a faulty state to a normal state spontaneously.

2)-b): It follows from the construction process of $L(G^o_{\log})$ from G. A circuit in $L(G^o_{\log})$ where all state labels are equal to N corresponds to a circuit in G that is only reached by a normal string. A circuit in $L(G^o_{\log})$ where all state labels are equal to F corresponds to a circuit in G that is reached by a string containing at least one fault event. ■

Lemma 3: NFA G_v and NFA $G^o_{\log}$ have the following relationship. Consider an elementary circuit in G_v

$$\pi = \left(q_v^1, e_1, q_v^2\right)\left(q_v^2, e_2, q_v^3\right) \cdots \left(\left(q_v^n, e_n, q_v^1\right)\right.$$

with $q_v^k = ((q_k^1, f_k^1), (q_k^2, f_k^2)) \in Q_v$ and $e_k \in E_v$ for $k = 1, 2, \ldots, n$. Assume that q_v^1 is reached by $\omega \in L(G_v)$ from an initial state in G_v, then we have the following.

1) There must exist two elementary circuits in $G^o_{\log}$, i.e.,

$$\pi_1 = \left(\left(q_1^1, f_1^1\right), e_1, \left(q_2^1, f_2^1\right)\right)\left(\left(q_2^1, f_2^1\right), e_2, \left(q_3^1, f_3^1\right)\right) \cdots \left(\left(q_n^1, f_n^1\right), e_n, \left(q_1^1, f_1^1\right)\right)$$

$$\pi_2 = \left(\left(q_1^2, f_1^2\right), e_1, \left(q_2^2, f_2^2\right)\right)\left(\left(q_2^2, f_2^2\right), e_2, \left(q_3^2, f_3^2\right)\right) \cdots \left(\left(q_n^2, f_n^2\right), e_n, \left(q_1^2, f_1^2\right)\right)$$

such that (q_1^1, f_1^1) and (q_1^2, f_1^2) are both reached by ω from an initial state of $G^o_{\log}$.

2) $f_1^1 = f_2^1 = \cdots = f_n^1$ and $f_1^2 = f_2^2 = \cdots = f_n^2$.

Proof: 1) A circuit π in NFA G_v corresponds to two circuits in $L(G^o_{\log})$, since G_v is the parallel composition of $G^o_{\log}$ and $G^o_{\log}$. More precisely, the first elements of all states of circuit π form a circuit in $L(G^o_{\log})$, and the second elements of all states of circuit π form another circuit in $L(G^o_{\log})$.

2) We assume that $f_1^1 = f_2^1 = \cdots = f_n^1$ does not hold. That is, there exist f_i^1 and f_j^1 satisfying that $f_i^1 = N$ and $f_j^1 = F$, $i \neq j$ and $i, j \in \{1, 2, \ldots, n\}$. According to 1) of this lemma, there exists a circuit in $G^o_{\log}$ where the labels of two states are different, which contradicts the results presented in Lemma 2. Hence, we have $f_1^1 = f_2^1 = \cdots = f_n^1$. Similarly, $f_1^2 = f_2^2 = \cdots = f_n^2$ holds. ■

Now a necessary and sufficient condition based on G_v is proposed for the diagnosability of UMPA G.

Theorem 1: A UMPA G is diagnosable with respect to projection $P : E^* \to E_o^*$ and set of faults E_f iff for every node $((q_v^1, f^1), (q_v^2, f^2)) \in Q_v$ that belongs to an elementary circuit in G_v, we have $f^1 = f^2$.

Proof (If): We assume that for every node $((q_v^1, f^1), (q_v^2, f^2)) \in Q_v$ that belongs to an elementary circuit in G_v, we have $f^1 = f^2$. This means that for any state $((q^1, f^1), (q^2, f^2)) \in Q_v$, if $f^1 \neq f^2$, then $((q^1, f^1), (q^2, f^2))$ is not contained in an elementary circuit of G_v. Besides, since G_v is finite, then the above implies that the length of any path $\pi = (q_v^1, e_1, q_v^2)(q_v^2, e_2, q_v^3) \cdots (q_v^{n-1}, e_{n-1}, q_v^n)$ containing only uncertain states $q_v^k = ((q_k^1, f_k^1), (q_k^2, f_k^2))$, $k = 1, 2, \ldots, n$, is bounded by $|Q_v| - 1$, i.e., $n \leq |Q_v| - 1$.

Let $\gamma \in L(G)$ be a timed sequence ending with a fault event $e_f \in E_f$, i.e., $lab_l(\gamma) = e_f$. From any state $x_v \in Q_v$ that is reachable from an initial state by $P(\gamma)^{\text{elem}} \in E_v^*$ in G_v, a state

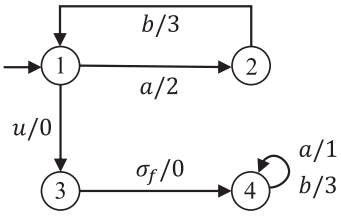

Fig. 5. Unambiguous max-plus automaton G.

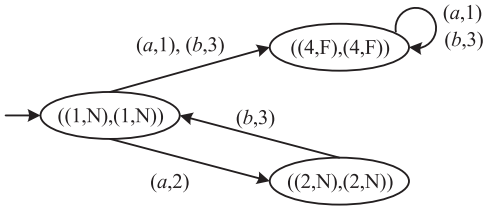

Fig. 6. Diagram of G_v of G in Fig. 5.

$y_v = ((y_v^1, f^1), ((y_v^1, f^2)) \in Q_v$ with $f^1 = f^2$ can be reached within $|Q_v| - 1$ steps. Then, we can conclude that for any $v = \gamma t \in L(G)$ with $|P(t)| \geq |Q_v| - 1$ and any $\omega \in L(G)$ such that $P(\omega) = P(v)$, ω must contain a fault event that belongs to E_f. Otherwise, there must exist elementary circuits that only contain uncertain states, which contradicts the assumption. Therefore, according to Definition 7 on diagnosability, G is diagnosable.

(Only If): We suppose that there exists an uncertain state $((q_v^1, f^1), (q_v^2, f^2)) \in Q_v$, i.e., $f^1 \neq f^2$, such that it belongs to an elementary circuit in G_v. According to Lemma 3, this assumption implies that there exists at least one elementary circuit consisting only of uncertain states in G_v. Suppose that path $\pi = (q_v^1, e_1, q_v^2)(q_v^2, e_2, q_v^3) \cdots ((q_v^n, e_n, q_v^1)$, with $q_v^k = ((q_k^1, f_k^1), (q_k^2, f_k^2)) \in Q_v$ and $e_k \in E_v$ for $k = 1, 2, \ldots, n$, is such a circuit. We can further assume that $q_v^1 = ((q_1^1, f_1^1), (q_1^2, f_1^2))$ is reached by $v \in L(G_v)$ from an initial state in G_v. Then according to Lemma 3, we know that there must exist two elementary circuits in $G_{\log}^o$, i.e.,

$$\begin{aligned}\pi_1 = &\left(\left(q_1^1, f_1^1\right), e_1, \left(q_2^1, f_2^1\right)\right)\left(\left(q_2^1, f_2^1\right), e_2, \left(q_3^1, f_3^1\right)\right)\\ &\cdots\left(\left(q_n^1, f_n^1\right), e_n, \left(q_1^1, f_1^1\right)\right)\\ \pi_2 = &\left(\left(q_1^2, f_1^2\right), e_1, \left(q_2^2, f_2^2\right)\right)\left(\left(q_2^2, f_2^2\right), e_2, \left(q_3^2, f_3^2\right)\right)\\ &\cdots\left(\left(q_n^2, f_n^2\right), e_n, \left(q_1^2, f_1^2\right)\right)\end{aligned}$$

such that $f_1^1 = f_2^1 = \cdots = f_n^1 = N$ and $f_1^2 = f_2^2 = \cdots = f_n^2 = F$. Let $s = e_1 e_2 \cdots e_n$. Then, by Lemma 2, there must exist two timed sequences $\gamma_1, \gamma_2 \in L(G)$ such that $P(\gamma_1)^{\text{elem}} = P(\gamma_2)^{\text{elem}} = vs^j$, γ_1 is fault-free, and γ_2 contains a fault event. Since j can be an arbitrary integer, by Definition 7 on diagnosability, G is not diagnosable. ■

Example 4: Let us consider again UMPA G presented in Fig. 1. The NFA G_v of G is presented in Fig. 4. Since node $\{(2, N), (4, F)\}$ belongs to the elementary circuit $\{(2, N), (4, F)\} \xrightarrow{(b,3)} \{(1, N), (4, F)\} \xrightarrow{(a,2)} \{(2, N), (4, F)\}$ in G_v, by Theorem 1, G is not diagnosable.

Example 5: Consider UMPA G in Fig. 5, where $E_o = \{a, b\}$, $E_{uo} = E_f \cup E_{\text{reg}}$ with $E_f = \{\sigma_f\}$ and $E_{\text{reg}} = \{u\}$. After applying Algorithm 2, we obtain NFA G_v shown in Fig. 6. By Theorem 1, G is diagnosable.

Remark 2: By Algorithm 2, the number of states of NFA $G_{\log}^o = (Q^o, E_o^{log}, \delta_o, Q_i^o)$ for a UMPA $G = (Q, E, t, \varrho, Q_i)$, in the worst case, is equal to the number of states in augmented automaton G^A, i.e., $|Q^o| = |Q^A| = 2|Q|$. This implies that the number of states of NFA G_v is at most $|Q_v| = 4|Q|^2$ since G_v is the parallel composition of $G_{\log}^o$ and $G_{\log}^o$. Besides, the number of transitions in G_v is at most $|Q_v| \times |E_v| \times |Q_v|$, where $|E_v|$ is bounded by $|Q|^3 \times |E_o|$. The necessary and sufficient condition in Theorem 1 can be verified by finding all strongly connected components in G_v, whose complexity is $\mathcal{O}(4|Q|^2 + 4|Q|^2 \times (|Q|^3 \times |E_o|) \times 4|Q|^2) = \mathcal{O}(|Q|^7)$. Therefore, the diagnosability of a UMPA G can be verified with a polynomial-time complexity.

Remark 3: For a diagnosable system G, according to the (If) part of Theorem 1, we know that for any timed sequence $\gamma t \in L(G)$ where γ ends with a fault event $e_f \in E_f$, if $|P(t)| \geq |Q_v| - 1$, then we can certainly infer the occurrence of fault. In addition, due to the assumption that no circuits are labeled only by unobservable events in G, there may be up to $|Q| - 1$ unobservable events between any two consecutive observable events in $P(t)$. That is, we have $|t| \leq (|P(t)|+1) \times (|Q|-1)$, i,e., $|P(t)| \geq |t|/(|Q| - 1) - 1$. Therefore, if $|t| \geq |Q_v| \times (|Q| - 1)$, then $|P(t)| \geq |t|/(|Q| - 1) - 1 \geq (|Q_v| \times (|Q| - 1))/(|Q| - 1) - 1 = |Q_v| - 1$. In other words, a fault in diagnosable system G can be detected with a delay of at most $|Q_v| \times (|Q| - 1) = 4|Q|^2 \times (|Q| - 1) = 4|Q|^3 - 4|Q|^2$ event occurrences after it occurred.

The polynomial approach proposed in this article is probably not optimal, because already in the logical case there are approaches, e.g., [6], that considerably improve the complexity with respect to the approach in [5] on which our extension to UMPAs is based. One perspective of this work is therefore to revisit this first approach on the diagnosability of MPAs, taking inspiration from the contributions in [6] on logic automata, with the aim of optimizing the complexity. It should also be noted that our approach extends the work in [23] which studies the diagnosability of timed DESs represented by weighted automata over a finite horizon (i.e., satisfying either the acyclicity assumption or the assumption that all circuits are zero weight).

VI. From Diagnosability to T-Diagnosability

In this section, we show that T-diagnosability of a UMPA G can be analyzed by the diagnosability. More precisely, we first prove that G is T-diagnosable if and only if it is diagnosable. Then, for a diagnosable UMPA G, an algebraic way is proposed to calculate the maximum time required to detect the occurrence of a fault.

Proposition 1: A UMPA G is T-diagnosable if and only if it is diagnosable.

Proof (If): Assume that UMPA G is diagnosable. This means that there exists an integer $K \in \mathbb{N}$ such that the occurrence of a fault can be detected within the occurrence of K events after the fault has occurred. For each path of length K leading from

a state that has an input transition labeled by a fault event in G, we calculate the sum of its transition weights. Since each weight in G is a real number, the sum of transition weights of a path of length K is always finite. Let T be the maximum of these sums. Then, after a fault has occurred, the fault must be detected within up to T time units, i.e., G is T-diagnosable.

(Only If): It is equivalent to prove that if G is not diagnosable, then it is not T-diagnosable. Assume that G is not diagnosable. According to the definition of diagnosability, we have

$$\begin{aligned}&(\forall K \in \mathbb{N})\big(\exists s \in L(G), lab_l(s) \in E_f\big)\\&(\exists v = st \in L(G), |t| \geq K)\\&\quad \Rightarrow (\exists \omega \in L(G), P(\omega) = P(v))\big(\forall e_f \in E_f, e_f \notin lab(\omega)\big).\end{aligned}$$

In other words, for any $K \in \{1, 2, \ldots, \}$, there exist two timed sequences $s_1^K = s\sigma^K \in L(G)$, with $|\sigma^K| \geq K$, and $s_2^K \in L(G)$ such that they have the same projection, s_1^K contains a fault event (note that s ends with a fault), and s_2^K does not contain any fault. If K converges to ∞, then $|s_1^K| = |s\sigma^K|$ converges to ∞. In addition, due to Assumption A4, i.e., there is no circuit with a sum of transition weights equal to 0, $t_l(s\sigma^K)$ converges to ∞. The above implies that for any $T \in \mathbb{R}$, there exist two timed sequences $st \in L(G)$, with $t_l(st) - t_l(s) \geq T$, and $\omega \in L(G)$ such that the projections of st and ω are the same, s ends with a fault event, and ω is fault-free. Formally

$$\begin{aligned}&(\forall T \in \mathbb{R})\big(\exists s \in L(G), lab_l(s) \in E_f\big)\\&(\exists v = st \in L(G), t_l(st) - t_l(s) \geq T)\\&\quad \Rightarrow (\exists \omega \in L(G), P(\omega) = P(v))\big(\forall e_f \in E_f, e_f \notin lab(\omega)\big).\end{aligned}$$

That is, G is not T-diagnosable. ∎

For any subset $Q_s \subseteq Q$ in a UMPA $G = (Q, E, t, \varrho, Q_i)$, we define a vector denoted by $\alpha^s \in \mathbb{R}_{\max}^{1\times|Q|}$ with respect to Q_s as

$$\alpha_q^s = \begin{cases} 0, & \text{if } q \in Q_s \\ \varepsilon, & \text{otherwise.}\end{cases}$$

Note that α_q^s represents the qth element in α^s. We further define column vector $I = (0, 0, \ldots, 0)^T \in \mathbb{R}_{\max}^{|Q|\times 1}$ with all elements equal to 0, and matrix $M^K = [\bigoplus_{e\in E} \mu(e)]^K$, where $\mu(e) \in \mathbb{R}_{\max}^{|Q|\times|Q|}$ with $\mu(e)_{pq} = t(p, e, q)$, and M^K is defined as $M^K = \overbrace{M \otimes M \otimes \ldots \otimes M}^{K \text{ times}}$.

Lemma 4: The maximum of the sums of the transition weights of all paths of length K leading from states in Q_s to a state in UMPA G is given by $\alpha^s \otimes M^K \otimes I$.

Proof: According to the definition of $\oplus$ and $\otimes$ specified in Section III, we know that value M_{pq}^K represents the maximum of the sums of transition weights of all paths of length K leading from state p to q in G. By combining the definitions of α^s, I, and operation $\otimes$, it is clear that $\alpha^s \otimes M^K \otimes I$ corresponds to the maximum weights of all paths of length K leading from states in Q_s to a state in G. ∎

Now we present an algebraic way to calculate the maximum time within which the occurrence of any fault can be detected in a diagnosable UMPA G. Let $Q_f = \{q \in Q | \exists p \in Q, \exists e_f \in E_f : t(p, e_f, q) \neq \varepsilon\}$ be the set of states having an input transition that is labeled by a fault event in G. Similar to α^s, we define vector $\alpha^f \in \mathbb{R}_{\max}^{1\times|Q|}$ with respect to Q_f as

$$\alpha_q^f = \begin{cases} 0, & \text{if } q \in Q_f \\ \varepsilon, & \text{otherwise.}\end{cases}$$

Note that α_q^f represents the qth element in α^f. Then, according to the (if) part of Proposition 1 and Lemma 4, we know that the maximal time (denoted by T) needed to detect the occurrence of any fault in a diagnosable UMPA G is given by

$$T = \alpha^f \otimes M^K \otimes I \tag{2}$$

where $K = 4|Q|^3 - 4|Q|^2$, as explained in Remark 3.

Example 6: Consider again the UMPA G in Fig. 5, where $E_o = \{a, b\}$, $E_{uo} = E_f \cup E_{\text{reg}}$ with $E_f = \{f\}$ and $E_{\text{reg}} = \{u\}$. It has been shown in Example 5 that G is diagnosable with a bound $K = |Q_v| \times (|Q| - 1) = 9$. In this example, we have $\alpha^f = (\varepsilon, \varepsilon, \varepsilon, 0)$, $I = (0, 0, 0, 0)$, and

$$M^9 = \begin{pmatrix} \varepsilon & 22 & 20 & 21 \\ 23 & \varepsilon & \varepsilon & 21 \\ \varepsilon & \varepsilon & \varepsilon & 24 \\ \varepsilon & \varepsilon & \varepsilon & 27 \end{pmatrix}.$$

By (2), we obtain an upper bound on the time to detect the occurrence of fault f, which is equal to $T = \alpha^f \otimes M^9 \otimes I = 27$.

Remark 4: According to (2), we know that the complexity of computing the maximum time to detect the fault occurrence for a diagnosable UMPA (Q, E, t, ϱ, Q_i) is polynomial in the number of states of G since the multiplication of matrices has a complexity that is polynomial in the dimension of the matrices. More precisely, by (2), it is easy to check that the complexity is equal to $\mathcal{O}(|Q|^2 + (K - 1) \times |Q|^3 + |Q|) = \mathcal{O}(|Q|^2 + (4|Q|^3 - 4|Q|^2 - 1) \times |Q|^3 + |Q|) = \mathcal{O}(|Q|^6)$.

VII. Conclusion

The diagnosability and T-diagnosability for UMPAs are studied in this article. We propose a polynomial-time algorithm based on the construction of an NFA over a weighted alphabet for diagnosability verification of UMPAs. Note that unlike [23], we do not assume that the underlying logical automaton is diagnosable, which means that the time information is fully useful to improve the diagnosability of the timed systems compared to the logical cases. In addition, we introduce an approach to calculate the maximum time for detecting fault occurrence for a diagnosable UMPA, and its complexity is of sixth order in the number of states of the UMPA. It is likely that (interval) T-time PNs under strong firing semantics can be encoded by deterministic weighted automata with weights in an interval semiring (direct product of max-plus semiring with itself). We can then have polynomial fault diagnosis algorithms for fairly large classes of timed systems. In the future, it is worthwhile to search for new techniques to check the diagnosability and T-diagnosability for general MPAs. Another possible extension is to study co-diagnosability in the decentralized setting or modular fault diagnosis for MPAs.

REFERENCES

[1] J. Zaytoon and S. Lafortune, "Overview of fault diagnosis methods for discrete event systems," *Annu. Rev. Control*, vol. 37, no. 2, pp. 308–320, 2013.

[2] S. Lafortune, F. Lin, and C. N. Hadjicostis, "On the history of diagnosability and opacity in discrete event systems," *Annu. Rev. Control*, vol. 45, pp. 257–266, Apr. 2018.

[3] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Diagnosability of discrete-event systems," *IEEE Trans. Autom. Control*, vol. 40, no. 9, pp. 1555–1575, Sep. 1995.

[4] T.-S. Yoo and S. Lafortune, "Polynomial-time verification of diagnosability of partially observed discrete-event systems," *IEEE Trans. Autom. Control*, vol. 47, no. 9, pp. 1491–1495, Sep. 2002.

[5] S. Jiang, Z. Huang, V. Chandra, and R. Kumar, "A polynomial algorithm for testing diagnosability of discrete-event systems," *IEEE Trans. Autom. Control*, vol. 46, no. 8, pp. 1318–1321, Aug. 2001.

[6] M. V. Moreira, T. C. Jesus, and J. C. Basilio, "Polynomial time verification of decentralized diagnosability of discrete event systems," *IEEE Trans. Autom. Control*, vol. 56, no. 7, pp. 1679–1684, Jul. 2011.

[7] K. W. Schmidt, "Verification of modular diagnosability with local specifications for discrete-event systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 5, pp. 1130–1140, Sep. 2013.

[8] M. Dotoli, M. P. Fanti, A. M. Mangini, and W. Ukovich, "On-line fault detection in discrete event systems by Petri nets and integer linear programming," *Automatica*, vol. 45, no. 11, pp. 2665–2672, 2009.

[9] F. Basile, P. Chiacchio, and G. De Tommasi, "On k-diagnosability of Petri nets via integer linear programming," *Automatica*, vol. 48, no. 9, pp. 2047–2058, 2012.

[10] X. Cong, M. P. Fanti, A. M. Mangini, and Z. Li, "Decentralized diagnosis by Petri nets and integer linear programming," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 10, pp. 1689–1700, Oct. 2018.

[11] M. P. Cabasino, A. Giua, and C. Seatzu, "Diagnosability of discrete-event systems using labeled Petri nets," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 1, pp. 144–153, Jan. 2014.

[12] X. Yin and S. Lafortune, "On the decidability and complexity of diagnosability for labeled Petri nets," *IEEE Trans. Autom. Control*, vol. 62, no. 11, pp. 5931–5938, Nov. 2017.

[13] C. Mahulea, C. Seatzu, M. P. Cabasino, and M. Silva, "Fault diagnosis of discrete-event systems using continuous Petri nets," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 42, no. 4, pp. 970–984, Jul. 2012.

[14] S. Tripakis, "Fault diagnosis for timed automata," in *Proc. Int. Symp. Formal Techn. Real Time Fault Tolerant Syst.*, 2002, pp. 205–221.

[15] P. Bouyer, F. Chevalier, and D. D'Souza, "Fault diagnosis using timed automata," in *Proc. Int. Conf. Found. Softw. Sci. Comput. Struct.*, 2005, pp. 219–233.

[16] S. H. Zad, R. H. Kwong, and W. M. Wonham, "Fault diagnosis in discrete-event systems: Incorporating timing information," *IEEE Trans. Autom. Control*, vol. 50, no. 7, pp. 1010–1015, Jul. 2005.

[17] J. Pan and S. Hashtrudi-Zad, "Diagnosability test for timed discrete-event systems," in *Proc. 18th IEEE Int. Conf. Tools Artif. Intell. (ICTAI)*, 2006, pp. 63–72.

[18] B. A. Brandin and W. M. Wonham, "Supervisory control of timed discrete-event systems," *IEEE Trans. Autom. Control*, vol. 39, no. 2, pp. 329–342, Feb. 1994.

[19] F. Basile, M. P. Cabasino, and C. Seatzu, "Diagnosability analysis of labeled time Petri net systems," *IEEE Trans. Autom. Control*, vol. 62, no. 3, pp. 1384–1396, Mar. 2017.

[20] G. S. Viana, J. C. Basilio, and M. V. Moreira, "Computation of the maximum time for failure diagnosis of discrete-event systems," in *Proc. IEEE Amer. Control Conf. (ACC)*, 2015, pp. 396–401.

[21] R. Su, J. H. Van Schuppen, and J. E. Rooda, "The synthesis of time optimal supervisors by using heaps-of-pieces," *IEEE Trans. Autom. Control*, vol. 57, no. 1, pp. 105–118, Jan. 2012.

[22] G. S. Viana and J. C. Basilio, "Codiagnosability of discrete event systems revisited: A new necessary and sufficient condition and its applications," *Automatica*, vol. 101, pp. 354–364, Mar. 2019.

[23] G. S. Viana, M. V. Moreira, and J. C. Basilio, "Codiagnosability analysis of discrete-event systems modeled by weighted automata," *IEEE Trans. Autom. Control*, vol. 64, no. 10, pp. 4361–4368, Oct. 2019.

[24] S. Gaubert, "Performance evaluation of (max,+) automata," *IEEE Trans. Autom. Control*, vol. 40, no. 12, pp. 2014–2025, Dec. 1995.

[25] J. Komenda, S. Lahaye, and J.-L. Boimond, "Determinization of timed Petri nets behaviors," *Discr. Event Dyn. Syst.*, vol. 26, no. 3, pp. 413–437, 2016.

[26] L. Triska and T. Moor, "Behaviour equivalent max-plus automata for a class of timed Petri nets," *IFAC-PapersOnLine*, vol. 53, no. 4, pp. 75–82, 2020.

[27] L. Triska and T. Moor, "Behaviour equivalent max-plus automata for timed Petri nets under open-loop race-policy semantics," *Discr. Event Dyn. Syst.*, vol. 31, no. 4, pp. 583–607, 2021.

[28] F. Cassez, "The complexity of codiagnosability for discrete event and timed systems," *IEEE Trans. Autom. Control*, vol. 57, no. 7, pp. 1752–1764, Jul. 2012.

[29] R. Alur and D. L. Dill, "A theory of timed automata," *Theor. Comput. Sci.*, vol. 126, no. 2, pp. 183–235, 1994.

[30] G. Viana, M. V. S. Alves, and J. C. Basilio, "Codiagnosability of networked discrete event systems with timing structure," *IEEE Trans. Autom. Control*, early access, Aug. 30, 2021, doi: 10.1109/TAC.2021.3108518.

[31] A. Lai, S. Lahaye, and Z. Li, "Initial-state detectability and initial-state opacity of unambiguous weighted automata," *Automatica*, vol. 127, May 2021, Art. no. 109490.

[32] M. Droste, W. Kuich, and H. Vogler, *Handbook of Weighted Automata*. New York, NY, USA: Springer, 2009.

[33] F. Baccelli, G. Cohen, G. J. Olsder, and J.-P. Quadrat, *Synchronization and Linearity: An Algebra for Discrete Event Systems*. Hoboken, NJ, USA: Wiley, 1992.

Aiwen Lai received the B.S. degree in automation from Xidian University, Xi'an, China, in 2014, the M.S. degree in information systems science from Aix-Marseille University, Marseille, France, in 2016, and the Ph.D. degree in automatic control from the University of Angers, Angers, France, in 2019.

He is currently an Assistant Professor with the Department of Automation, Xiamen University, Xiamen, China. His current research interests include state estimation, fault diagnosis, and opacity of discrete-event systems.

Jan Komenda received the M.Sc. degree in mathematics from Masaryk University, Brno, Czech Republic, in 1994, and the Ph.D. degree in control and computer science from the Université de Franche-Comté, Besançon, France, in 1999.

He worked with CWI, Amsterdam, The Netherlands, from 2001 to 2003 in the Group of J. H. van Schuppen. He is a Researcher with the Institute of Mathematics, Czech Academy of Sciences, Brno. His current research interests are in supervisory control of logical and timed discrete-event systems using methods of coalgebra and idempotent algebra.

Dr. Komenda serves as an Associate Editor for IEEE TRANSACTIONS ON AUTOMATIC CONTROL and the Department Editor for *Discrete Event Dynamic Systems* and has served as an Associate Editor for *Automatica* from 2009 to 2015.

Sébastien Lahaye received the Ph.D. degree in systems and computer engineering from the University of Angers, Angers, France, in 2000.

He is a Professor of Automatic Control with Polytech Angers, University of Angers. His research interests include discrete-event systems, complex systems, control, properties analysis and verification, max-plus algebra, Petri nets, and automata.