Timed Fault Diagnosis in Switching Output Automata

Tianyu Liu, Carla Seatzu and Alessandro Giua

Abstract— **In this paper, we study the problem of fault diagnosis in Switching Output Automata (SOA), which is a formal modeling framework particularly suitable for describing artificial systems with discrete or quantized piecewise continuous outputs. We introduce Switching Output Automata with Faults (SOAF), an extension of SOA that incorporates timed fault arcs to model fault occurrences under specific temporal constraints. In our framework, faults are defined with respect to both the discrete states and the system outputs, where each fault occurrence is constrained to specific time intervals determined by the current global state of the system. To perform fault diagnosis, we introduce the Evolution Automaton with Faults (EAF), a nondetermistic finite automaton that serves as a logical abstraction of the SOAF. Based on the approach of Sampath and Lafortune, we construct a diagnoser for EAF that effectively performs fault diagnosis in the SOAF framework.**

I. INTRODUCTION

System faults are typically unexpected events triggered by abnormalities in one or more system components. These faults are often difficult to predict, can result in significant financial losses, and can even lead to catastrophic consequences. To reduce the impact of system faults, a structured process is required to guarantee their prompt identification. Within the framework of Discrete Event Systems (DES), faults are defined as discrete event-driven transitions that trigger abnormal system behaviours. Such a formalisation enables systematic fault diagnosis through discrete state analysis [1]. Several models and methods for fault diagnosis of DES have been proposed in the literature [1,2,3,4].

In our work, we focus on a novel model structure, SOA, proposed and further studied in [5,6,7]. This is an intuitive formalization method that is well-suited for describing artificial systems. In these systems, the output can take discrete values or quantized piecewise continuous values. We introduce new arcs in the SOA to represent faults, and faults may only occur within specified time intervals. The time intervals in which faults may occur depend on the global state, namely, the discrete state and the output. In other words, starting from the same discrete state, faults may occur at different time intervals depending on the current value of the output.

In this paper, we provide the following contributions. First, we define the notion of *timed fault* for SOA and define the *switching output automaton with faults* to include the faulty behavior in the model. Second, we define the *evolution automaton with faults* as a nondetermistic finite automaton to provide a purely discrete abstraction of the behavior of the system subject to faults. Finally, we adapt the diagnoser framework of Sampath et al. [3] to our EAF, enabling effective fault diagnosis for systems modeled as SOAF.

II. PRELIMINARIES

In this section, we introduce SOA, with details provided in [5,6,7].

Definition 1: A *switching output automaton* is defined as $G = (X, Y, B, h, x_0, y_0)$, where X is a finite set of states; Y is an *output alphabet*; $B \subseteq X \times X$ is a set of arcs (or edges); $h : X \to 2^Y$ is the *output function*; $x_0 \in X$ is the initial state; $y_0 \in h(x_0)$ is the initial output symbol.

We define $h(x) \subseteq Y$ as the set of potential output symbols that can be generated when the current state is x. An arc $b = (x, x') \in B$ is considered to be directed from state x to state x'. We define the set of direct successors of x as $\sigma(x) = \{x' \in X | \exists b = (x, x') \in B\}$.

A state-output run describes the evolution of an SOA. Runs consist of three types of transitions. Type 1 refers to a state change with no output change. Type 2 indicates a simultaneous state and output change. Type 3 indicates an output change with no state change. To avoid *Zeno phenomena*, we assume that the time distance between the occurrence of any two such transitions must be greater than or equal to the minimum dwell time δ.

A global state of the SOA is a pair $q = (x, y) \in X \times Y$ such that $y \in h(x) \subseteq Y$. The set of global states is denoted Q. The number of global states is $n_Q = |Q| = \sum_{x \in X} |h(x)| \leq |X| \cdot |Y|$.

We define $Q_x = \{(x, y) | y \in h(x)\} \subseteq Q$ as the set of global states associated with state x. For an arc $b = (x, x') \in B$, $q = (x, y) \in Q_x$ is a global state associated with state x and $q' = (x', y') \in Q_{x'}$ is a global state associated with state x', then among the possible evolutions (runs) of the SOA, there exists a run including transition $q \xrightarrow{t} q'$ where t indicates when the transition occurs. Clearly, in the evolution of the SOA, the number of transitions generated by the arc $b = (x, x')$ is $|h(x)| \cdot |h(x')|$.

The system's observation (output behavior) is a mapping from time $T = \mathbb{R}_{\geq 0}$ (where $\mathbb{R}_{\geq 0}$ denotes the set of non-negative real numbers) to output Y. The output $y(t)$ is piecewise continuous, with each value lasting at least δ time units. The system state, a piecewise continuous function $x : T \to X$, cannot be directly measured but can be estimated from the system model and the output evolution.

This work is partially supported by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU and the MIUR Project (PRIN), call 2022, ERC Sector: PE6, for the project "Motown: Smart Production Planning and Control for Manufacturing of Electric Vehicle Powertrain in Industry 4.0 Environment" (CUP:20228XEHRE).

Tianyu Liu, Carla Seatzu and Alessandro Giua DIEE, University of Cagliari, Cagliari 09123, Italy (e-mail: t.liu@studenti.unica.it; carla.seatzu@unica.it; giua@unica.it).

III. Problem Statement

In this work, we consider the problem of timed fault diagnosis for SOA. Failures in real-world engineering systems are often not random occurrences, but are triggered under specific operating conditions and accumulated time (such as physical phenomena like overheating, wear, voltage fluctuations, etc.). This time dependency is crucial for building more reliable fault diagnosis systems.

We assume that when the system is in certain specific states, within specific time periods, it is prone to failures. Failures may lead the system to deviate from the expected (nominal) evolution and enter a state that should not have been reached. States from which faults may occur are called *fault-prone states*. The set of fault-prone states is denoted $X_f \in X$. Moreover, after a fault occurs, the system can continue to operate without becoming paralyzed.

To formalize the occurrence of faults, we add some *fault arcs* to the SOA, whose starting nodes belong to X_f and whose endpoints belong to X. Unlike the arcs in B, fault arcs are only active at certain time intervals. The set of fault arcs is denoted as $B_f \subseteq (X_f \times X)$ and it is $B_f \cap B = \varnothing$. Just like an arc $b \in B$, a fault arc $b_f = (x,x') \in B_f$ can generate multiple transitions, indeed in general, both the starting and the ending nodes correspond to different global states being $|h(x)|$ and $|h(x')| \geq 1$. We define the set of direct successors of x reachable via a fault transition as $\sigma_f(x) = \{x' \in X | \exists b_f = (x,x') \in B_f\}$.

We further assume that the dwell time of the fault arc $b_f = (x,x')$ is closely related to the output of its starting state $x \in X_f$. When the system is in state x, the time window for system failures depend on the current output. We define Q_f as the set of global states of all fault-prone states X_f, referred to as *fault-prone global states*. Furthermore, we assume that the system can only experience a fault after the minimal dwell time has elapsed, and each fault may occur within different time intervals. In more detail, a *fault time mapping* $\mathcal{I}$ associates with a global state $q = (x,y) \in Q_f$ (where $x \in X_f$ and $y \in h(x)$) and its corresponding fault arc $b_f = (x,x')$ a set of *fault occurrence intervals*

$$\mathcal{I}(q,b_f) = \{[\delta'_1,\delta''_1),[\delta'_2,\delta''_2),...,[\delta'_n,\delta''_n)\}$$

where $n \in \mathbb{N}_+$ and each interval end precedes the start of the next one, i.e., $\delta''_m < \delta'_{m+1}$, $1 \leq m < n$. The above intervals depend on the output value y of the global state q and the fault arc b_f. Specifically:

- If multiple outputs $y_1, y_2, \ldots$ are possible at state x, each output may generate distinct time intervals through mapping $\mathcal{I}$.
- A single output y may correspond to multiple non-consecutive intervals (e.g., $[\delta'_1,\delta''_1)$ and $[\delta'_2,\delta''_2)$) when faults can be triggered under the same output due to:
  - Multiple Triggering Rules: Independent physical phenomena (e.g., overheating and voltage spikes) imposing separate time constraints.

We assume that the lower and upper bounds of each interval are multiples of the minimum dwell time δ, i.e.,

$$\delta'_m = k'_m\delta, \delta''_m = k''_m\delta, \ 1 \leq m \leq n.$$

It is worth noting that for all $1 \leq m \leq n$, it is $k'_m \in \mathbb{N}_+$; for all $1 \leq m < n$, it is $k''_m \in \mathbb{N}_+$; finally, for $m = n$, it is $k''_m \in \mathbb{N}_+ \cup \{+\infty\}$. Using the minimum dwell time δ as our fundamental time unit creates a discrete representation that effectively captures the system's temporal evolution. This discretization approach forms the basis for the EAF construction detailed in Section IV.

Definition 2: Given a SOA $G = (X,Y,B,h,x_0,y_0)$, a set of *fault arcs* B_f, a set of *fault-prone global states* Q_f, and $\mathcal{I}$: $Q_f \times B_f \to 2^{\mathbb{R}\times(\mathbb{R}\cup\{+\infty\})}$ is the *fault time mapping*. The corresponding SOAF is defined as $G_f = \langle G,B_f,Q_f,\mathcal{I}\rangle$.

We can still use state-output runs to describe the evolution of a SOAF, where the evolution incorporates fault occurrences in addition to the standard SOA evolution. The SOAF's observation (output behavior) is still a mapping from time $T = \mathbb{R}_{\geq 0}$ to output Y. We use (y,τ) to denote that the output takes value y for a time interval of duration τ where $y \in Y$ and $\tau \in \mathbb{R}_{\geq 0}$. The output behavior of G_f is defined as $L(G_f) = \{\omega \in (Y \times \mathbb{R}_{\geq\delta})^+ | \forall i = 1,2,\ldots,n : \omega = (y_{[0,\tau_1)},\tau_1)(y_{[\tau_1,\tau_2)},\tau_2 - \tau_1)\cdots(y_{[\tau_{i-1},\tau_i)},\tau_i - \tau_{i-1})$ and $y_{[\tau_{k-1},\tau_k)} \neq y_{[\tau_k,\tau_{k+1})}, k = 1,\ldots,i-1$ and $\tau_i - \tau_{i-1} \geq \delta\}$.

Example 1: Consider the SOA $G = (X,Y,B,h,x_0,y_0)$ in Fig. 1, with states set $X = \{x_0,x_1,x_2\}$, output alphabet $Y = \{1,2\}$, arcs set $B = \{(x_0,x_1),(x_0,x_2)\}$, output function defined by $h(x_0) = \{1\}, h(x_1) = \{1,2\}$, $h(x_2) = \{1\}$, initial state x_0 and initial output symbol $y_0 = 1$. The set of direct successors of x_0 is $\sigma(x_0) = \{x_1,x_2\}$.

We define the set of fault arcs as $B_f = \{(x_1,x_0),(x_1,x_2)\}$ and mark them in red. Consequently, we have $X_f = \{x_1\}$ and $Q_f = \{(x_1,1),(x_1,2)\}$. We set the following fault occurrence intervals:

- $\mathcal{I}((x_1,1),(x_1,x_0)) = \{[\delta,2\delta)\}$;
- $\mathcal{I}((x_1,2),(x_1,x_0)) = \{[\delta,2\delta)\}$;
- $\mathcal{I}((x_1,2),(x_1,x_2)) = \{[2\delta,3\delta)\}$.

The SOAF G_f is shown in Fig. 2. We label the fault arc (x_1,x_0) as $'1:\{[\delta,2\delta)\},2:\{[\delta,2\delta)\}'$ to indicate that for state x_1 with an output value of 1, the fault occurrence interval is $[\delta,2\delta)$, and with an output value of 2, the fault occurrence interval is $[\delta,2\delta)$. We label the fault arc (x_1,x_2) as $'2:\{[2\delta,3\delta)\}'$ to indicate that for state x_1 with an output value of 2, the fault occurrence interval is $[2\delta,3\delta)$.

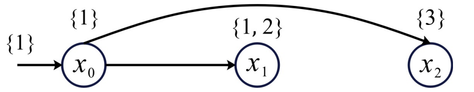

Fig. 1: A switching output automaton.

IV. Fault diagnosis

Fault diagnosis involves detecting the occurrence of faults by analyzing the output evolution. To make fault diagnosis on a *switching output automaton with faults*, we propose a four-step structured approach: (1) Construction of Logical Global

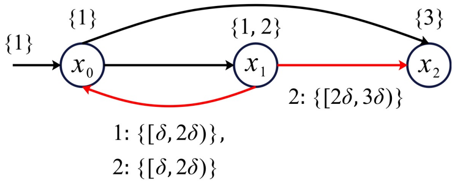
{1}
{1}
{1, 2}
{3}
x_0
x_1
x_2
2: $\{[2\delta,3\delta)\}$
1: $\{[\delta,2\delta)\}$,
2: $\{[\delta,2\delta)\}$

Fig. 2: The SOAF associated with the SOA in Fig. 1 and defined in Example 1.

States, (2) Construction of the EAF, (3) Construction of the Fault Recognizer, and (4) Construction of the Diagnoser. Subsections IV.A to IV.D illustrate these steps in detail.

A. Construction of Logical Global States

According to the definition of SOAF, we partition the set of possible dwell times associated with a global state into suitable intervals.

The logical global state $(x,y)_j$ denotes a condition in which the global state is $q=(x,y)$ and its dwell time is in the interval I_j. Here δ denotes the minimum dwell time. We define a transition labeled δ (referred to as a δ−transition) to represent the logical event: a period of time equal to the minimum dwell time has elapsed. $Q_f \subseteq Q$ is the set of fault-prone global states and B_f is a set of fault arcs(as defined in Definition 3).

- If $q=(x,y)\notin Q_f$, i.e., q is not prone to failure, intervals of interest are:

$$I_0=[0,\delta) \quad \text{and} \quad I_1=[\delta,+\infty)$$

leading to the logical sequence

$$(x,y)_0 \xrightarrow{\delta} (x,y)_1 \ \circlearrowleft \ \delta$$

- If $q=(x,y)\in Q_f$ is a fault-prone global state, there may exist multiple fault occurrence intervals for q due to the presence of multiple fault arcs originating from x and each arc potentially being triggerable in multiple time intervals.

We define $B_{f,x}\subseteq B_f$ as fault arcs originating from x and assume there exist r fault arcs in $B_{f,x}$, i.e., $B_{f,x}=\{b^1_{f,x},b^2_{f,x},...,b^r_{f,x}\}$. For each fault arc in $B_{f,x}$, the corresponding fault occurrence intervals for the fault-prone global state q are defined as follows:

$$\mathcal{I}(q,b^1_{f,x})=\{[k'_{1,1}\delta,k''_{1,1}\delta),...,[k'_{n,1}\delta,k''_{n,1}\delta)\}$$

$$\mathcal{I}(q,b^2_{f,x})=\{[k'_{1,2}\delta,k''_{1,2}\delta),...,[k'_{n,2}\delta,k''_{n,2}\delta)\}$$

$$\vdots$$

$$\mathcal{I}(q,b^r_{f,x})=\{[k'_{1,r}\delta,k''_{1,r}\delta),...,[k'_{n,r}\delta,k''_{n,r}\delta)\}.$$

Let k be defined as

$$k=\begin{cases}\max\{k''_{n,1},k''_{n,2},\ldots,k''_{n,r}\}, \text{ if } \max\{k''_{n,1},\ldots,k''_{n,r}\}\in\mathbb{N}_+\\ \max\{k''_{n,s}\mid k''_{n,s}\neq+\infty,\ 1\leq s\leq r\}, \text{ otherwise}\end{cases} \tag{1}$$

where $\mathbb{N}_+$ denotes the set of positive integers. In the first case, k takes the maximum value when it is a positive integer; otherwise, it adopts the maximum finite value from the set after excluding $+\infty$.

In such a case, intervals of interest are: $I_0=[0,\delta),I_1=[\delta,2\delta),\ldots,I_{k-1}=[(k-1)\delta,k\delta)$ and $I_k=[k\delta,+\infty)$ corresponding to logical sequence

$$(x,y)_0 \overset{\delta}{\to} (x,y)_1 \overset{\delta}{\to} \ldots \overset{\delta}{\to} (x,y)_{k-1} \overset{\delta}{\to} (x,y)_k \circlearrowleft \delta.$$

We define a function $\mathcal{R}:Q\to\mathbb{N}_+$ that maps each global state $q\in Q$ to the maximum interval index associated with its dwell-time partitioning. The function is formally given by:

$$\mathcal{R}(q)=\begin{cases}1, & \text{if } q\notin Q_f\\ k, & \text{if } q\in Q_f\end{cases} \tag{2}$$

where Q_f is a set of fault-prone global states and k is defined in equation (1).

B. Construction of the Evolution Automaton with Faults

We construct a nondeterministic automaton called EAF, denoted as G_{ef}. The EAF is an extension of the *secret-dependent evolution automaton* introduced in [6], which provides a purely logical abstraction for the evolutionary process of the SOA by quantifying the dwell time of global states, enabling system behavior modeling and analysis using finite state automata. The EAF refines this model by incorporating both fault transitions and normal transitions (time-driven and regular transitions), thereby enabling a comprehensive analysis of SOAF behavior.

To simplify our analysis, we restrict the alphabet to just two symbols $\{n,f\}$, where n represents normal transitions and f represents fault transitions. In the following subsections, we delineate the distinct types of transitions that constitute the EAF framework.

1) Time-driven transitions: Time-driven transitions represent evolutionary processes where no change occurs in the discrete state or output of the system. Correspondingly, in the EAF, we represent a time-driven evolution as a sequence of logical global states. Formally, a time-driven transition labeled n from the logical global state $(x,y)_j$ yields the logical global state $(x,y)_{j+1}$ if $j<\mathcal{R}((x,y))$, otherwise it results in a self-loop in $(x,y)_j$.

2) Regular transitions: Regular transitions describe the changes in the discrete state or output of the system.

From logical global state $(x,y)_j$,

- when $j=0$, the state and output cannot change until the minimum dwell time elapses;
- when $j>0$, changes of output and transitions to a different discrete state are possible.

There are three types of transitions that may occur:

a) Type 1 (state change with no output change): When the system is in global state (x,y) and within time interval I_j, if a Type 1 transition occurs, causing the discrete state to change from x to $\bar{x}\in\sigma(x)\neq x$ while the output y remains unchanged, then externally it's impossible to detect the occurrence of this transition based on the output. Mathematically, this can be represented as:

$$(x,y)_j \xrightarrow{n} (\bar{x},y)_0$$

where the duration of stay in $(\bar{x},y)$ starts from zero and n indicates that this transition is modeling a normal behavior.

b) Type 2 (simultaneous state and output change): When the system is in global state (x,y) and within time interval I_j, if a Type 2 transition occurs, causing both the discrete state to change from x to $\bar{x} \in \sigma(x) \neq x$ and the output to change from y to $\bar{y} \in h(\bar{x}) \neq y$, then a new observable output is produced. Mathematically, this can be represented as:

$$(x,y)_j \xrightarrow{n} (\bar{x},\bar{y})_0$$

where the duration of stay in $(\bar{x},\bar{y})$ starts from zero and n indicates again a normal behavior.

c) Type 3 (output change with no state change): When the system is in global state (x,y) and within time interval I_j, if a Type 3 transition occurs, causing the output to change from y to $\bar{y} \in h(x) \neq y$ while the discrete state x remains unchanged, then a new observable output is produced. Mathematically, this can be represented as:

$$(x,y)_j \xrightarrow{n} (x,\bar{y})_0$$

where the duration of stay in $(x,\bar{y})$ starts from zero and n indicates normal behavior.

3) Fault transitions: For a fault-prone global state (x,y) and a fault arc $b^s_{f,x} \in B_{f,x}$ (where $1 \leq s \leq r$) starting from x, logical global states $(x,y)_j$ is called a *fault-prone logical global state* if the interval I_j belongs to an interval in $\mathcal{I}(q,b^s_{f,x})$. It corresponds to conditions in which the SOAF is in a global state $q=(x,y)$ with a dwell time t such that when t belongs to an interval in $\mathcal{I}(q,b^s_{f,x})$, the system can trigger a fault arc $b^s_{f,x}$, thereby leading to a fault.

We define the function $\mathcal{T}$, which takes a set of time intervals $\mathcal{I}(q,b^s_{f,x})$ as input, and outputs the union of the coefficient pairs of these intervals. The function is defined as follows:

$$\mathcal{T}(\mathcal{I}(q,b^s_{f,x})) = \begin{cases} \bigcup_{i=1}^{n}[k'_{i,s},k''_{i,s}), \text{ if } k''_{n,s} \neq +\infty \\ \bigcup_{i=1}^{n-1}[k'_{i,s},k''_{i,s}) \cup [k'_{n,s},k], \text{ otherwise} \end{cases} \tag{3}$$

where k is defined in equation (1). The above function allows to determine for which values of j, $(x,y)_j$ are fault-prone logical global states.

When the system is in global state (x,y) and within time interval $I_j \in \mathcal{I}((x,y),(x,x'))$, if a fault transition occurs following a fault arc $b_f=(x,x')$, the system moves to a new discrete state x' and the output may either change to $y' \in h(x') \neq y$ or remain unchanged $(y'=y)$, depending on all possible outputs $h(x')$. If the output changes, such a transition is observable; if the output remains unchanged, it cannot be detected by an external observer. Mathematically, this can be represented as:

$$(x,y)_j \xrightarrow{f} (x',y')_0$$

where the duration of stay in (x',y') starts from zero, $y' \in h(x')$, and f indicates that this transition is a fault transition.

Definition 3: Given an SOAF $G_f = \langle G, B_f, Q_f, \mathcal{I} \rangle$, its EAF is a nondeterministic finite automaton $G_{ef} = (Q_e, Y_e, \Delta, q_0)$ where

- $Q_e = \{(x,y)_j | q=(x,y) \in Q, j \in \{0,\cdots,\mathcal{R}(q)\}$ is a finite set of logical global states;
- $Y_e = \{n, f\}$ is the set of events;
- $\Delta \subseteq Q_e \times Y_e \times Q_e$ is the *transition relation*;
- $q_0 = (x_0,y_0)_0$ is the initial state.

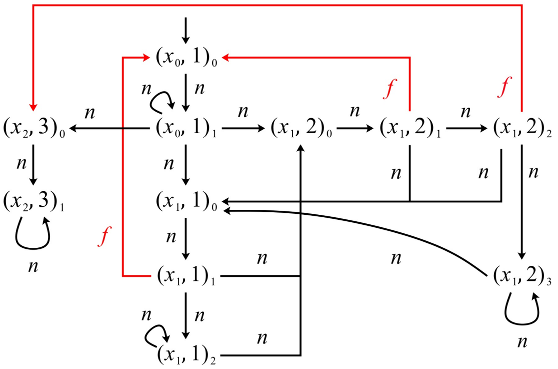

Fig. 3: The evolution automaton with faults of the switching output automaton with faults in Fig. 2.

Example 2: Consider the SOAF shown in Fig. 2, The set of fault-prone global states $Q_f = \{(x_1,1),(x_1,2)\}$. $\mathcal{T}(\mathcal{I}((x_1,1),(x_1,x_0))) = \{1\}$, $\mathcal{T}(\mathcal{I}((x_1,2),(x_1,x_0))) = \{1\}$ and $\mathcal{T}(\mathcal{I}((x_1,2),(x_1,x_2))) = \{2\}$. The set of fault-prone logical global states is $\{(x_1,1)_1,(x_1,2)_1,(x_1,2)_2\}$. The EAF G_{ef} is shown in Fig. 3.

C. Construction of the Fault Recognizer

The EAF completely enumerates all possible runs of the original SOAF. Each fault or normal evolution in the SOAF is faithfully captured by an equivalent path in the EAF. Therefore, if a fault can (or cannot) occur in the SOAF, the same possibility is reflected in the transitions of the EAF. To implement fault diagnosis, we can construct a diagnoser for the evolution automaton with faults according to the theoretical framework proposed by Sampath and Lafortune [9].

Definition 4: Given an EAF $G_{ef} = (Q_e, Y_e, \Delta, q_0)$ with alphabet Y_e and set of fault events $Y_f \subset Y_e$, a *fault monitor* is the deterministic finite automaton $M = (X_M, Y_e, \delta_M, x_{M,0})$ where $X_M = \{N, F\}$ is the set of states, $x_{M,0} = N$ is the initial state, Y_e is the alphabet, the transition function $\delta_M : X_M \times Y_e \to X_M$ is defined as:

$$\delta_M(x, y_e) = \begin{cases} N & \text{if } x = N \text{ and } y_e = n \\ F & \text{otherwise.} \end{cases} \tag{4}$$

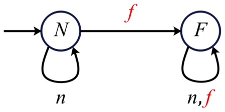

Fig. 4: The fault monitor M for the EAF in Fig. 3.

The *fault recognizer* for G_{ef} is the nondeterministic finite automaton $Rec(G_{ef}) = G_{ef} \parallel M$ obtained by the concurrent composition of G_{ef} and M. It generates language

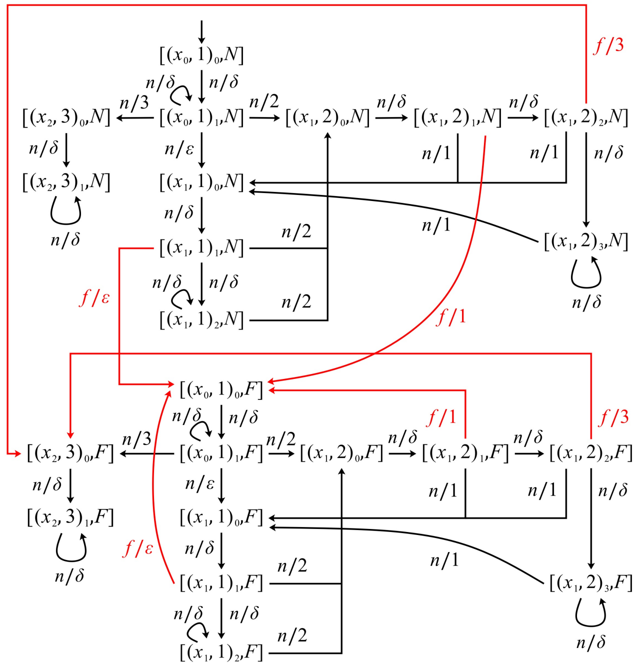

Fig. 5: The fault recognizer with observation label function for the EAF in Fig. 3.

$L(Rec(G_{ef})) = L(G_{ef})$ and has the following structure: $Rec(G_{ef}) = (X_R, Y_e, \Delta_R, x_{R,0})$ where

- the state set is $X_R \subseteq Q_e \times \{N,F\}$;
- the initial state is $x_{R,0} = (q_0, N)$;
- the transition relation is $\Delta_R \subseteq X_R \times Y_e \times X_R$.

We observe that each transition relation $(x_r^1, y_e, x_r^2) \in \Delta_R$ encapsulates all information regarding the system's discrete state changes, output variations, and fault occurrences. External observers can only monitor the system's outputs and their durations. Therefore, we can assign an observation label to each transition relation to elucidate the specific system changes occurring when a particular output is observed.

We define a function $\mathcal{P} : \Delta_R \to Y \cup \{\delta\} \cup \{\varepsilon\}$ that maps each transition relation to its corresponding observation label, where elements in Y represent observable outputs, δ denotes an observable minimal dwell time, and ε represents unobservable events. The function is formally defined as:

$$\mathcal{P}((x_r^1, y_e, x_r^2)) = \begin{cases} y_2, & \text{if } y_1 \neq y_2 \\ \delta, & \text{if } (x_1,y_1) = (x_2,y_2) \\ \varepsilon, & \text{if } y_1 = y_2 \text{ and } (x_1,y_1) \neq (x_2,y_2) \end{cases} \quad (5)$$

where $x_r^1 = [(x_1,y_1)_{j_1}, \gamma_1]$ and $x_r^2 = [(x_2,y_2)_{j_2}, \gamma_2]$. We can extend $\mathcal{P} : \Delta_R^* \to \{Y \cup \{\delta\} \cup \{\varepsilon\}\}^*$ which satisfies the following properties: $\mathcal{P}(\varepsilon) = \varepsilon$ and $\mathcal{P}(\alpha\omega) = \mathcal{P}(\alpha) \cdot \mathcal{P}(\omega)$, where $\alpha \in \Delta_R$ and $\omega \in \Delta_R^*$.

Example 3: Consider the EAF $G_{ef} = (Q_e, Y_e, \Delta, q_0)$ illustrated in Fig. 3. The corresponding fault monitor is depicted in Fig. 4, while Fig. 5 presents the fault recognizer with its associated observation label function.

In order to perform fault diagnosis on the SOAF, we need to abstract the output behaviors of the SOAF into logical observations. The output behaviors of the SOAF have already been defined in Section III.

Definition 5: Given a SOAF $G_f = \langle G, B_f, Q_f, \mathcal{I} \rangle$, let $L(G_f)$ be the set of output behaviors. We define function $\psi : L(G_f) \to (Y \cup \{\delta\})^*$ as follows:

$$\psi((y,t)) = \delta^k \text{ with } k = \lfloor t/\delta \rfloor \text{ and}$$

$$\psi(\omega(y,t)) = \psi(\omega)\ y\delta^k \text{ with } k = \lfloor t/\delta \rfloor$$

where $\lfloor \cdot \rfloor$ denotes the floor function and $(y,t), \omega, \omega(y,t) \in L(G_f)$.

Example 4: Consider the SOAF in Fig. 2. Let the minimum dwell time be $\delta = 1$ and the output behavior $\omega' = (1,3)(2,5.4)(1,3.9)$. We compute the corresponding sequence $u = \psi(\omega') = \psi((1,3)(2,5.4))1\delta^3 = \psi((1,3))2\delta^5 1\delta^3 = \delta^3 2\delta^5 1\delta^3$.

D. Construction of diagnoser

Thus we can construct the equivalent deterministic automaton of the fault recognizer, called *diagnoser (observer)* $G_{diag} = (Z, Y \cup \{\delta\}, \Delta_o, z_0)$ where

- $Z \subseteq 2^{X_R}$ is a finite state set, i.e., each state of the diagnoser is a set of pairs $z = \{(q_{e1},\gamma_1),(q_{e2},\gamma_2),\ldots,(q_{ek},\gamma_k)\}$, where $q_{ei} \in Q_e$ and $\gamma_i \in \{N,F\}$, for $i = 1,2,\ldots,k$.
- $Y \cup \{\delta\}$ is the alphabet;
- $\Delta_o : Z \times (Y \cup \{\delta\}) \to Z$ is the partial transition function;
- $z_0 = \{(q_0, N)\}$ is the initial state.

The observer can be used to estimate the set of states that are consistent with any logical observation sequence $u = \psi(\omega)$ where $\omega \in L(G_f)$.

For any observation state $z \in Z$ of automaton G_{diag}, where $z = \{(q_{e1},\gamma_1),(q_{e2},\gamma_2),\ldots,(q_{ek},\gamma_k)\}$, we define the diagnostic evaluation function $\varphi : Z \to \{N,F,U\}$ as follows:

1) The state z is classified as normal ($\varphi(z) = N$) when: $\forall i \in \{1,2,\ldots,k\} : \gamma_i = N$
2) The state z is classified as faulty ($\varphi(z) = F$) when: $\forall i \in \{1,2,\ldots,k\} : \gamma_i = F$
3) The state z is classified as uncertain ($\varphi(z) = U$) when: $\exists (i,j) \in \{1,2,\ldots,k\}^2 : \gamma_i = N \wedge \gamma_j = F$

Proposition 1: Given a SOAF $G_f = \langle G, B_f, Q_f, \mathcal{I} \rangle$ and its diagnoser G_{diag}, let $\omega \in L(G_f)$ be a sequence of output behaviors, $u = \psi(\omega)$ be its corresponding logical observation under mapping ψ, and $z = \Delta_o(z_0, u)$ be the set of states reached in G_{diag} after observing u. The fault diagnosis can be characterized as follows:

1) no fault has occurred if $\varphi(z) = N$;
2) a fault is detected if $\varphi(z) = F$;
3) a fault is uncertain if $\varphi(z) = U$.

Proof: When $\phi(z) = N$, all possible states in the diagnoser are labeled as normal, so no fault has occurred in the system; when $\phi(z) = F$, all possible states are labeled as faulty, so a fault has been detected in the system; when $\phi(z) = U$, there exists a mixture of normal and faulty labels, so the fault status is uncertain. Since the diagnoser state $z = \Delta_o(z_0, u)$ contains all possible system states and their fault labels consistent with the observation sequence u, the proposition holds.

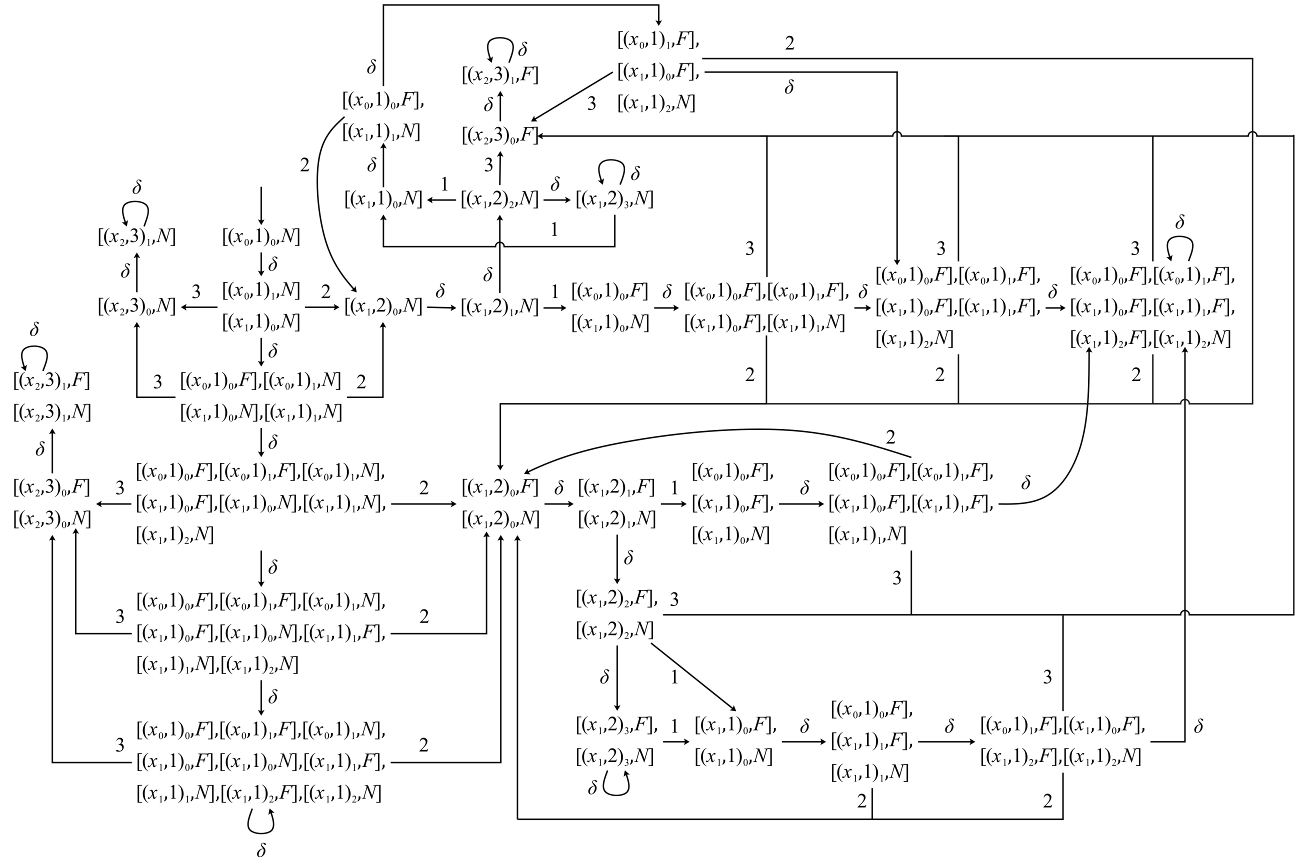
[(x₀,1)₁,F], [(x₁,1)₀,F], [(x₁,1)₂,N]
[(x₂,3)₁,F]
[(x₂,3)₀,F]
[(x₀,1)₀,F], [(x₁,1)₁,N]
[(x₁,1)₀,N]
[(x₁,2)₂,N]
[(x₁,2)₃,N]
[(x₂,3)₁,N]
[(x₂,3)₀,N]
[(x₀,1)₀,N]
[(x₀,1)₁,N] [(x₁,1)₀,N]
[(x₁,2)₀,N]
[(x₁,2)₁,N]
[(x₀,1)₀,F] [(x₁,1)₀,N]
[(x₀,1)₀,F],[(x₀,1)₁,F], [(x₁,1)₀,F],[(x₁,1)₁,N]
[(x₀,1)₀,F],[(x₀,1)₁,F], [(x₁,1)₀,F],[(x₁,1)₁,F], [(x₁,1)₂,N]
[(x₀,1)₀,F],[(x₀,1)₁,F], [(x₁,1)₀,F],[(x₁,1)₁,F], [(x₁,1)₂,F],[(x₁,1)₂,N]
[(x₀,1)₀,F],[(x₀,1)₁,N] [(x₁,1)₀,N],[(x₁,1)₁,N]
[(x₂,3)₁,F] [(x₂,3)₁,N]
[(x₂,3)₀,F] [(x₂,3)₀,N]
[(x₀,1)₀,F],[(x₀,1)₁,F],[(x₀,1)₁,N], [(x₁,1)₀,F],[(x₁,1)₀,N],[(x₁,1)₁,N], [(x₁,1)₂,N]
[(x₁,2)₀,F] [(x₁,2)₀,N]
[(x₁,2)₁,F] [(x₁,2)₁,N]
[(x₀,1)₀,F], [(x₁,1)₀,F], [(x₁,1)₀,N]
[(x₀,1)₀,F],[(x₀,1)₁,F], [(x₁,1)₀,F],[(x₁,1)₁,F], [(x₁,1)₁,N]
[(x₀,1)₀,F],[(x₀,1)₁,F],[(x₀,1)₁,N], [(x₁,1)₀,F],[(x₁,1)₀,N],[(x₁,1)₁,F], [(x₁,1)₁,N],[(x₁,1)₂,N]
[(x₁,2)₂,F], [(x₁,2)₂,N]
[(x₀,1)₀,F],[(x₀,1)₁,F],[(x₀,1)₁,N], [(x₁,1)₀,F],[(x₁,1)₀,N],[(x₁,1)₁,F], [(x₁,1)₁,N],[(x₁,1)₂,F],[(x₁,1)₂,N]
[(x₁,2)₃,F], [(x₁,2)₃,N]
[(x₁,1)₀,F], [(x₁,1)₀,N]
[(x₀,1)₀,F], [(x₁,1)₁,F], [(x₁,1)₁,N]
[(x₀,1)₁,F],[(x₁,1)₀,F], [(x₁,1)₂,F],[(x₁,1)₂,N]
δ
1
2
3

Fig. 6: The diagnoser for the EAF in Fig. 3.

■

The computational complexity of the entire algorithm is $O(2^{2|Q_e|})$, which grows exponentially.

Example 5: Consider the EAF illustrated in Fig. 3 and its corresponding representations: the fault monitor (Fig. 4), the fault recognizer (Fig. 5), and the diagnoser (Fig. 6).

Let us analyze two distinct output behaviors: $\omega_1 = (1,1.5)(3,5.4)$ and $\omega_2 = (1,1.9)(2,2.3)(3,0.8)$. The corresponding logical observation sequences derived from these behaviors are $u_1 = \psi(\omega_1) = \delta 3 \delta^5$ and $u_2 = \psi(\omega_2) = \delta 2 \delta^2 3$, respectively. By applying these logical observation sequences to our diagnoser, the consistent states reached are:

$$z_1 = \Delta_o(z_0, u_1) = \{[(x_2,3)_1, N]\}$$

$$z_2 = \Delta_o(z_0, u_2) = \{[(x_2,3)_0, F]\}.$$

Consequently, we obtain $\varphi(z_1) = N$ and $\varphi(z_2) = F$.

Based on this analysis, we can draw definitive conclusions about the system's fault status: when output ω_1 is observed, no fault has occurred; when output ω_2 is observed, a fault has indeed occurred.

V. CONCLUSIONS AND FUTURE WORKS

In this paper, we have presented a framework for diagnosing timed faults in SOA by introducing SOAF and EAF formalisms. Building upon the classical diagnoser approach pioneered by Sampath and Lafortune, we have developed an effective diagnostic solution for systems with discrete or quantized piecewise continuous outputs. Our framework rigorously handles fault occurrences under specific temporal constraints, providing a sound theoretical foundation for fault detection and identification.

Although the discretization of SOAF evolutions preserves all necessary information for diagnosability analysis, we have not demonstrated such analysis in the present work. As part of our future research, we plan to conduct a comprehensive diagnosability study and apply this diagnostic methodology to cyber-physical systems to validate its practical utility in real-world applications.

REFERENCES

[1] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems," *in Proceedings of the IEEE*, vol. 77, no. 1, pp. 81-98, 1989.

[2] C. G. Cassandras, and S. Lafortune, eds. *Introduction to discrete event systems.* Boston, MA: Springer US, 2008.

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

[4] C. N. Hadjicostis, Estimation and Inference in Discrete Event Systems: A Model-Based Approach with Finite Automata. *Springer*, 2020.

[5] T. Liu, C. Seatzu and A. Giua. Verification of Current State Opacity using Switching Output Automata. *CoDIT 2023*, pp. 2665-2670, 2023.

[6] T. Liu, C. Seatzu and A. Giua. Timed Opacity Verification for Switching Output Automata. *WODES 2024*, Volume 58, Issue 1, pp. 24-29, 2024.

[7] T. Liu, C. Seatzu, F. Pascucci, G. Cavone and A. Giua, Security-by-Design of Smart Water Supply Systems: a Switching Output Automaton-based Approach, *2024 CASE*, Bari, Italy, pp. 1532-1539, 2024.