Resilient Fault Diagnosis Under Imperfect Observations–A Need for Industry 4.0 Era

Alejandro White, *Member, IEEE*, Ali Karimoddini, *Senior Member, IEEE*, and Mohammad Karimadini

***Abstract*—In smart industrial systems, in many cases, a fault can be captured as an event to represent the distinct nature of subsequent changes. Event-based fault diagnosis techniques are capable model-based methods for diagnosing faults from a sequence of observable events executed by the system under diagnosis. Most event-based diagnosis techniques rely on perfect observations of observable events. However, in practice, it is common to miss an observable event due to a problem in sensor-readings or communication/transmission channels. This paper develops a fault diagnosis tool, referred to as diagnoser, which can robustly detect, locate, and isolate occurred faults. The developed diagnoser is resilient against missed observations. A missed observation is detected from its successive sequence of events. Upon detecting a missed observation, the developed diagnoser automatically resets and then, asynchronously resumes the diagnosis process. This is achieved solely based on post-reset/activation observations and without interrupting the performance of the system under diagnosis. New concepts of asynchronous detectability and asynchronous diagnosability are introduced. It is shown that if asynchronous detectability and asynchronous diagnosability hold, the proposed diagnoser is capable of diagnosing occurred faults under imperfect observations. The proposed technique is applied to diagnose faults in a manufacturing process. Illustrative examples are provided to explain the details of the proposed algorithm. The result paves the way towards fostering resilient cyber-physical systems in Industry 4.0 context.**

***Index Terms*—Cyber-physical systems, discrete event systems, fault diagnosis, imperfect communication, imperfect observation, Industry 4.0, resilience.**

I. Introduction

ADVANCES in technologies are revolutionizing traditional industries by an increasing shift toward integrated and distributed cyber-physical systems, in so-called Industry 4.0 era [1], where the complexity is moved from the mechanical structures to sensing, perception, planning, control, and decision-making components [2]–[4], and the priorities have shifted from pre-planned automation to reliable autonomy [5], [6]. Such increasingly complex engineered systems, such as industrial internet of things (IIOT) for manufacturing [7], require automatic diagnostic mechanisms with the capability to cipher through these system's complexities and provide a timely, clear, and concise diagnostic output that ensures reliable and safe system operations in order to achieve cyber-security [8]–[11].

Different diagnosis techniques include but are not limited to mathematical model based approaches [12]–[15], artificial intelligence techniques [16]–[21], fault tree analysis [22], [23], template structures [24], [25], model-checking [26], [27], Bayesian networks [28], and discrete event system (DES) methods [29]–[39]. Among these methods, DES approaches use time-abstract event-driven models of the systems under diagnosis and provide diagnostic information based on high-level logical behaviors of the systems, which is an effective strategy particularly when dealing with complex systems. Furthermore, DES models naturally capture faults as abrupt changes (events) in the system, which facilitates the analysis of faulty behaviors of the system. More importantly, the topology of a DES model is similar to the human cognitive process on correlating systems' interactions and the effect(s) of sequences of events [40], [41]. This makes DES framework very suitable for the decision-making layer of a control structure to manage normal/faulty situations toward a desired/safe sequence of events.

DES fault diagnosis has been applied to different systems including power transmission networks [42], automated manufacturing systems [43], communication networks [44], [45], cyber-security [46], and flight control systems [47]. In [48], an event-based diagnosis tool, so-called diagnoser, was developed. Using the abstraction of continuous dynamics of a system, an automaton-based fault detection and isolation technique was introduced in [35]. A state-based DES diagnosis technique was studied in [49]. In [50], a learning-based diagnosis technique is introduced for diagnosis of an unknown DES system, and in [51]–[53], an asynchronous diagnosis technique is developed, relaxing the generally required synchronous initialization between the diagnoser and the system under diagnosis. Once a fault is diagnosed, fault-accommodation techniques can be employed to recover the system [54]–[56].

All aforementioned DES fault diagnosis techniques rely on perfect observations of sequences of events executed by the

Manuscript received May 21, 2020; accepted June 25, 2020. This work was supported by the National Science Foundation (NSF) (1832110 and 2000320) and Air Force Research Laboratory (AFRL) and Office of the Secretary of Defense (OSD) (FA8750-15-2-0116). Recommended by Associate Editor Giuseppe Franzè. *(Corresponding author: Ali Karimoddini.)*

Citation: A. White, A. Karimoddini, and M. Karimadini, "Resilient fault diagnosis under imperfect observations–A need for Industry 4.0 era," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 5, pp. 1279–1288, Sept. 2020.

A. White is with Vehicle Technology Directorate, CCDC Army Research Laboratory, Aberdeen Proving Ground, MD 21005 USA (e-mail: alejandro.p.white2.civ@mail.mil).

A. Karimoddini is with the Department of Electrical and Computer Engineering, North Carolina Agricultural and Technical State University, Greensboro, NC 27411 USA (e-mail: akarimod@ncat.edu).

M. Karimadini is with the Department of Electrical Engineering, Arak University of Technology, Arak 3818146763, Iran (e-mail: karimadini@arakut.ac.ir).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JAS.2020.1003333

system under diagnosis. However, in practice, it is common that due to the problems in sensor-readings or communication/ transmission channels, an observation is missed. In these situations, the integrity of the observed sequence may lead to missed or improper diagnosis. This can result in the system with improper operation that it needs to switch out of, or erroneous execution of an incorrect recovery action. When multiple local diagnosers are available, [57] introduced a trace-based diagnosis process which can handle timing mismatch and channel distortion in a distributed setting. Reference [58] has addressed the problem of robust diagnosis, when diagnosers are themselves subject to failures, by taking the advantage of collective decision-making in a decentralized structure. In [59], a probabilistic method is developed for fault diagnosis, which captures the loss of communication/ observation as faulty events with a certain probability. An alternative solution to address the robust fault diagnosis problem is to consider the loss of observation of an event at a particular part of the model as a fault and treat it as an intermittent fault [60], [61] or as a permanent fault [62]. However, loss of observation may happen anytime anywhere, and considering an associated intermittent or permanent fault for the loss of observations for all events at different locations in the system will significantly make the system's analysis complex.

This paper addresses these challenges by proposing a novel event-based fault diagnosis technique which is resilient against missed observations. Here, the main difficulty is that when an observation is missed, the inference of the diagnosis being made based on subsequent observed events will be compromised. By now, the only solution in this situation is to restart the diagnosis process to track a valid sequence of events in the system under diagnosis. However, by resetting the diagnosis process, the past history of information about the system under diagnosis will be missed at the reset time, leaving us with a challenge to diagnose occurred faults based on post-reset/activation of the diagnoser. To tackle these problems, the proposed diagnoser automatically detects missed observations, resets, and then, resumes the diagnosis process, without interrupting the operation of the system under diagnosis. The new concepts of asynchronous detectability and diagnosability are introduced. It is also shown that if the asynchronous detectability and diagnosability hold, the developed diagnoser can detect the occurred faults under imperfect observations. The developed method is applied to the diagnosis of faults in a manufacturing system.

The rest of the paper is organized as follows. Section II provides the preliminaries and required definitions, descriptions, and notations utilized in the modeling and diagnosis of the DES systems. This section is concluded with a formal problem statement for resilient fault diagnosis. In Section III, the structure of the proposed diagnoser is explained followed by developing an algorithm for constructing the proposed resilient diagnoser. Section IV reviews some of the properties of the developed diagnoser. Section V derives the conditions for asynchronous diagnosability of occurred faults in a DES system under imperfect observations, and finally, Section VI concludes the paper.

II. Problem Formulation

Consider the system under diagnosis G, which is modeled as a non-deterministic finite-state automaton

$$G = (X, \Sigma, \delta, x_0) \tag{1}$$

where X is the state space of the system, $x_0 \in X$ is the system's initial state, Σ is the finite set of events, and $\delta : X \times \Sigma \to 2^X$ is the state transition relation. The event set Σ can be disjointly partitioned into the observable event set Σ_o and the unobservable event set Σ_u. A sequence of events forms a string. With the abuse of notation, $e \in t$ indicates that the event e is one of the events which form the string s. The length of a string t is shown by $|t|$. The concatenation of two strings s_1 and s_2 is shown by $s_1.s_2$. A set of strings forms a language. The set $pr(L)$ denotes the prefix closure of the language L, defined as $pr(L) = \{u \mid \exists v : u.v \in L\}$. The set $ext(L)$ is the extension closure of the language L, which can be defined as $ext(L) = \{v \mid \exists u \in L : u.v \in L\}$. The set of all possible finite strings over the set Σ, including the zero-length string ε, is shown by Σ^*.

We can extend the transition rule, δ, to a string in a recursive way as $\delta(x, \varepsilon) = x$ and $\delta(x, s.e) = \delta(\delta(x, s), e)$ for any $x \in X$, $s \in \Sigma^*$, and $e \in \Sigma$. The language of G can be defined as $L(G) = \{s \in \Sigma^* \mid \delta(x_0, s) \text{ is defined}\}$, which contains all strings that can be generated by the automaton G from the state x_0. $L(G)/s = \{t \in \Sigma^* \mid s.t \in L(G)\}$ includes the set of strings that G can generate after s, or namely $L(G)$ after s. The set $L(G(x)) = \{s \in \Sigma^* \mid \delta(x, s) \text{ is defined}\}$ referrers to the strings that can be generated by G from the state $x \in X$. The state $\tilde{x}$ can transit to $\tilde{x}'$ by the string s, shown by $\tilde{x} \xrightarrow{s} \tilde{x}'$, if $\tilde{x}' \in \delta(\tilde{x}, s)$. The string s can passes through $\tilde{x} \xrightarrow{\alpha} \tilde{x}'$ if there exist s_1, s_2, and $\tilde{x}''$ such that $x_0 \xrightarrow{s_1} \tilde{x} \xrightarrow{\alpha} \tilde{x}' \xrightarrow{s_2} \tilde{x}''$. The unobservable reach set $UR(x) = \{y \in X \mid \exists u \in \Sigma_u^*, \delta(x, u) = y\}$ includes the states that are reachable from state x by an unobservable string. The set $UE(s, x) = \{s.t \mid t \in \Sigma_u^* \text{ and } s.t \in L(G(x))\}$ contains all unobservable extensions of s that can be generated from the state x. For a string $t \in ext(L(G))$, $Pre(t) = \{s \mid s.t. \in L(G)\}$ includes the set of strings that can be the precedent of t. $Pre_a(t)$ is the actual precedent of t that has been actually executed by the plant before t.

System's faults can be captured by a set of unobservable events, $\Sigma_f \subseteq \Sigma_u$. Similar to [48], we consider m sets Σ_{f_1}, Σ_{f_2}, ..., Σ_{f_m}, each represents a particular fault type, where $\cup_{i=1}^{m} \Sigma_{f_i} = \Sigma_f$. A string $t \in L(G)$ is "*F_i-faulty*" if there exists an event $f \in \Sigma_{f_i}$, such that $f \in t$. A string $t \in L(G)$ is "*non F_i-faulty*" if for all $f \in \Sigma_{f_i}$, we have $f \notin t$. Finally, a string $t \in L(G)$ is "*normal*" if for all $f \in \Sigma_{f_i}$ and for all $i = 1, ..., m$, $f \notin t$.

The diagnosis problem includes detecting faults from the observable events executed by the system under diagnosis. The observable behavior of G can be captured by the natural projection to observable events, $P : \Sigma^* \to \Sigma_o^*$, which can be defined as follows:

1) $P(\varepsilon) = \varepsilon$;
2) $P(e) = e$, if $e \in \Sigma_o$;
3) $P(e) = \varepsilon$, if $e \notin \Sigma_o$;
4) $P(s.e) = P(s)P(e)$, for $s \in \Sigma^*$ and $e \in \Sigma$.

For a language $\mathcal{L}_1$, $P(\mathcal{L}_1) = \{P(s) \mid s \in \mathcal{L}_1\}$. The inverse

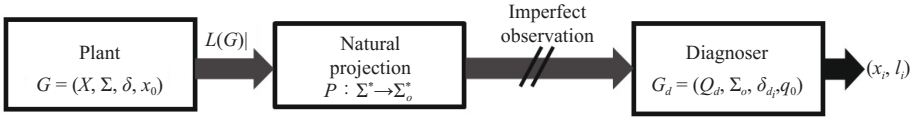

Fig. 1. The diagnosis process, in which the diagnoser should diagnose the occurred faults and their type, captured by fault label ℓ_i, and the location of the system under diagnosis, x_i, from imperfect observation of the system.

projection of a string $w \in \Sigma_o^*$ is $P^{-1}(w) = \{s \in L(G) \mid P(s) = w\}$.

Here, we assume that the system's language, $L(G)$, is live, i.e., $\forall x \in X, \exists \sigma \in \Sigma$ such that $\delta(x,\sigma)$ is defined. Further, we assume that the faults do not bring the system to a halt mode. These assumptions ensure that there is sufficient time to diagnose faults from observing the executed events. Further, we assume that the length of unobservable strings in $L(G)$ is bounded by n_o, ensuring that the system would not be stuck in a cycle of unobservable events, which is a reasonable assumption for live systems. On the other hand, we relax the commonly required assumption on perfect communication links or sensor readings, and hence, our formalism considers that some observable events may be missed in practice. The diagnosis problem is then to detect faults and determine their types and location based on observing the executed events. In the case of a missed observation, the successive events would not provide valid information until the diagnoser resets and resynchronizes with the system under diagnosis. This requires the diagnoser to detect faults based on post-reset/activation observations as described in the following problem:

Problem 1: From a run-time, imperfect, sufficiently large observation $P(t)$ of a DES system G, $t \in ext(L(G))$, determine if $\exists f \in \Sigma_f$ such that $f \in Pre_a(t).t$. If yes, identify the type of the occurred fault, Σ_{f_i}, where $f \in \Sigma_{f_i} \subseteq \Sigma_f$, and locate the fault by finding the system state $x \in X$ subsequently reached by $Pre_a(t).t$.

III. Constructing the Resilient Diagnoser

To address Problem 1, we introduce a diagnosis tool, called a diagnoser, which provides diagnostics by extracting information from the original system's observable events in order to estimate the original system's current state location and current condition (faulty or non-faulty) (See Fig. 1). The diagnoser can be represented by a finite-state automaton $G_{di} = (Q_d, \Sigma_d, \delta_{di}, q_0)$, where Q_d is the diagnoser's set of states, Σ_d is the diagnoser's event set, δ_{di} is the diagnoser's transition relation under imperfect observations, and q_0 is its initial state.

The diagnoser monitors the observable events of the system under diagnosis and changes its estimation (diagnoser state) accordingly. Therefore, the diagnoser's event set is the observable events, $\Sigma_d = \Sigma_o$. The states of the diagnoser $Q_d \subseteq 2^{X \times L}$ are in the form of $q_d = \{(x_1, \ell_1), \ldots, (x_k, \ell_k)\}$, in which the pairs $(x_j, \ell_j) \in q_d$ capture the estimation of the states of the system under diagnosis, $x_j \in X$, adjoined by their fault status, $\ell_j \subseteq L$. The set $L = \{N\} \cup 2^F$ contains *condition labels*, where $F = \{F_1, F_2, \ldots, F_m\}$ is the set of *fault labels*, in which F_i is the label for fault type Σ_{f_i}, $i = 1, \ldots, m$, and N is the label for normal system operation.

Definition 1 [48]: Consider $q = \{(x_1, \ell_1), \ldots, (x_M, \ell_M)\} \in Q_d$. Then, q is said to be

1) Normal if $\ell_k = \{N\}$ for all $k = 1, ..., M$.
2) F_i-certain if $F_i \in \ell_k$ for all $k = 1, ..., M$.
3) F_i-uncertain if $\exists n, m$ such that $F_i \in \ell_n$, but $F_i \notin \ell_m$.

The fault condition labels are tracked and propagated via the function $\nabla : L \times \Sigma^* \to L$ as

$$\nabla(\ell, t) = \begin{cases} \{N\}, \text{ if } \ell = \{N\} \text{ and } \forall f \in \Sigma_f,\ f \notin t, \\ \{F_i \in F | F_i \in \ell \text{ or } \exists f \in \Sigma_{fi},\ f \in t\}, \text{ otherwise.} \end{cases} \quad (2)$$

Algorithm 1 constructs the initial state of diagnoser, q_0, the diagnoser states, Q_d, and the diagnoser transition relation δ_{di}. In the first step, the algorithm constructs, q_0. The system under diagnosis is initially normal. However, during the diagnosis process, the diagnoser may reset any time due to missing an observation. Therefore, at the reset/activation instance, the diagnoser cannot assume that the system under diagnosis is normal. Instead, since after resetting the diagnoser, the past history of the system is not missed, the diagnoser should consider all states of the system and all their possible faulty statuses as the initial estimation of the system's state and status. Hence, the algorithm constructs q_0 by finding all states reachable from $(x_0, \{N\})$. For this purpose, the algorithm starts from $q_0 = \{(x_0, N)\}$, and then, extends q_0 to $x \in UR(x_0)$ by $q_0 = q_0 \cup \{(x, \ell) | x \in \delta(x_0, u),\ u \in \Sigma_u^*,\ \ell = \nabla(\{N\}, u)\}$. Then, the diagnoser will continue this process by searching over all other possible reachable states. To implement this idea, for each pair (x, ℓ) in q_0 and for each observable event e, the algorithm checks if (x, ℓ) can transit via e (and its unobservable extensions) to a new set of pairs

$$\bar{\delta}_d((x, \ell), e) := \{(y, \nabla(\ell, t)) | y \in \delta(x, t) \text{ and } t \in UE(e, x)\}. \quad (3)$$

The new identified pairs, $\bar{\delta}_d((x, \ell), e)$, are then included in q_0.

The second step of Algorithm 1 constructs the transition relation δ_{di} and the states of the diagnoser Q_d. Starting from q_0, the states of the diagnoser and its transition relation can be recursively constructed. For this purpose, for any $q \in Q_d$, the algorithm checks if for any $(x, \ell) \in q$ and for any $e \in \Sigma_o$, $\bar{\delta}_d((x, \ell), e)$ is defined. If so, $q' = \bigcup_{(x,\ell) \in q} \bar{\delta}_d((x, \ell), e)$ will be added to Q_d. At the same time, we add the transition from q to the new state q' to the list of admissible transitions of the diagnoser, which can happen when e is observed

$$\delta_{dp}(q, e) = \bigcup_{(x,\ell) \in q} \bar{\delta}_d((x, \ell), e). \quad (4)$$

With this information, we can construct $G_{dp} = (Q_d, \Sigma_o, \delta_{dp}, q_0)$ which can operate as a diagnoser under perfect observations. As it will be shown in Lemma 3, $ext(P(L(G))) = L(G_{dp})$, which means that if there is no missed observation, all transitions in $ext(P(L(G)))$ can be captured by $L(G_{dp})$. Given that, if for an event $e \in \Sigma_o$ and for all $(x, \ell) \in q$, $\bar{\delta}_d((x, \ell), e)$ is

not defined, but the diagnoser observes the event e when it is in state q, G_{dp} cannot recognize the event e, inferring that there has been a missed observation. To handle this situation, the diagnosis process should be restarted, by adding $\delta_{dr}(q,e)=q_0$ to the list of transitions. With this mechanism, we can form the diagnoser $G_{di}=(Q_d,\Sigma_o,\delta_{di},q_0)$ with $\delta_{di}=\delta_{dp}\cup\delta_{dr}$, which can operate under imperfect observations. If a missed observation is detected, the diagnoser G_{di} resets to q_0, and resumes the diagnosis process.

Algorithm 1 Constructing a Resilient Diagnoser

Input: $G=(X,\Sigma,\delta,x_0)$
Output: G_{dp}: the diagnoser under perfect observation
G_{di}: the diagnoser under imperfect observation
Initialization:
$q_0:=\{(x_0,N)\}$;
Step 1: Construct q_0
$q_0:=q_0\bigcup\{(x,\ell)\,|\,x\in\delta(x_0,u),\ u\in\Sigma_u^*,\ \ell=\nabla(\{N\},u)\}$;
repeat
for $(x,\ell)\in q_0$ and $e\in\Sigma_o$ **do**
if $\exists t\in UE(e,x)$ such that $y\in\delta(x,t)$ and $(y,\nabla(\ell,t))\notin q_0$ **then**
$\bar{\delta}_d((x,\ell),e):=\{(y,\nabla(\ell,t))|y\in\delta(x,t)$ and $t\in UE(e,x)\}$
$q_0=q_0\cup\bar{\delta}_d((x,\ell),e)$;
end if
end for
until there is no new pair (x,ℓ) in q_0
Step 2: Construct Q_d **and** δ_{di}
$Q_d:=\{q_0\}$,
repeat
for $q\in Q_d$ and $e\in\Sigma_o$ **do**
if $\exists(x,\ell)\in q$ s.t. $\bar{\delta}_d((x,\ell),e)$ is defined **then**
$\delta_{dp}(q,e)=\bigcup_{(x,\ell)\in q}\bar{\delta}_d((x,\ell),e)$;
add $\delta_{dp}(q,e)$ to Q_d;
else
$\delta_{dr}(q,e)=q_0$
end if
end for
until there is no new state $\delta_{dp}(q,e)$ for all $e\in\Sigma_o$.
$\delta_{di}=\delta_{dp}\cup\delta_{dr}$
$G_{dp}=(Q_d,\Sigma_o,\delta_{dp},q_0)$
$G_{di}=(Q_d,\Sigma_o,\delta_{di},q_0)$

Example 1: Consider a manufacturing system, a part of which conducts process β on the objects as shown in Fig. 2. For this purpose, the objects on the conveyor are pushed by the left pusher to the right station to conduct process β. In this station, there is a detector which evaluates if the object is processed well or it should be passed through process β again. Once the process is completed, the right pusher pushes the object back to the conveyor. The model of this manufacturing system is captured by automaton G_1 in Fig. 3, in which $\Sigma_o=\{\alpha,\beta\}$ contains the event α for observing the activation of the right and left pushers and the event β for conducting the process on the objects. As a part of initialization of the system, initially the process β is conducted two times to warm up the system, and then the pusher pushes the objects to be

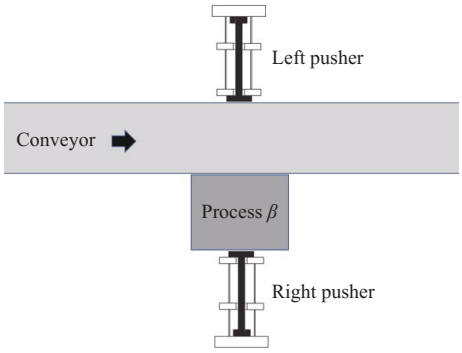

Fig. 2. The top view of a part of a manufacturing system: the objects on the conveyor are pushed by the left pusher to the right station to conduct process β. Once the detector confirms that the object is processed well, the process is completed and the right pusher pushes the object back to the conveyor.

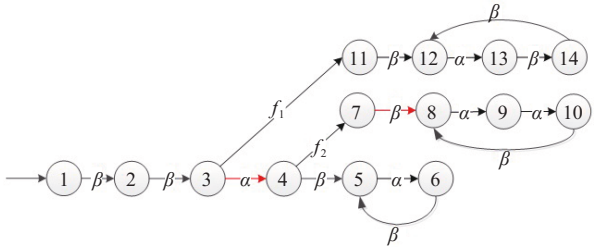

Fig. 3. The DES model of the manufacturing process described in Example 1, with the observable event set $\Sigma_o=\{\alpha,\beta\}$, which contains the event α for observing the activation of the right and left pushers and the event β for conducting the process on the objects. This system also include unobservable events $\Sigma_u=\Sigma_f=\{f_1,f_2\}$, with $\Sigma_{f_1}=\{f_1\}$, and $\Sigma_{f_2}=\{f_2\}$, in which f_1 stands for the fault in the detector resulting in the false positive detection of objects to be passed through process β, and f_2 stands for the fault in the right pusher when it fails to push the object back to the conveyor.

processed. To illustrate the proposed method, we consider two types of faults: f_1 which is a failure in the process β, and f_2, which is a failure in the pusher. These two fault types are captured by $\Sigma_u=\Sigma_f=\{f_1,f_2\}$, $\Sigma_{f_1}=\{f_1\}$, and $\Sigma_{f_2}=\{f_2\}$.

Applying Algorithm 1, the diagnoser G_{di} is constructed for G_1, as shown in Fig. 4. In the diagnoser's figure, to ease the drawings, instead of showing the states in the form of collections of pairs (x,ℓ), we have shown them simply as $x\ell$. For example, $q_1=\{(2,\{N\}),\ (3,\{N\}),\ (5,\{N\}),\ (8,\{F_2\}),\ (11,\{F_1\}),\ (12,\{F_1\}),\ (14,\{F_1\})\}$, is shown as $q_1=\{2N,3N,5N,8F_2,11F_1,12F_1,14F_1\}$. In this diagnoser's figure, the solid arrows represent δ_{dp} and the dashed arrows show the reset transitions, δ_{dr}.

Now consider the transition $1\xrightarrow{\beta}2\xrightarrow{\beta}3\xrightarrow{\alpha}4\xrightarrow{f_2}7\xrightarrow{\beta}8\xrightarrow{\alpha}9\xrightarrow{\alpha}10\xrightarrow{\beta}8\xrightarrow{\alpha}9\xrightarrow{\alpha}10$ in the system. Under perfect observation, the diagnoser observes $\beta\beta\alpha\beta\alpha\alpha\beta\alpha\alpha$ and executes the following sequence: $q_0\xrightarrow{\beta}q_1\xrightarrow{\beta}q_2\xrightarrow{\alpha}q_3\xrightarrow{\beta}q_4\xrightarrow{\alpha}q_5\xrightarrow{\alpha}q_9\xrightarrow{\beta}q_{10}\xrightarrow{\alpha}q_{11}\xrightarrow{\alpha}q_9$. As soon as the diagnoser reaches q_9 as an F_2-certain state, the diagnoser realizes that f_2 has occurred in the past.

Now, imagine that in the system's run $1\xrightarrow{\beta}2\xrightarrow{\beta}3\xrightarrow{\alpha}4\xrightarrow{f_2}7\xrightarrow{\beta}8\xrightarrow{\alpha}9\xrightarrow{\alpha}10\xrightarrow{\beta}8\xrightarrow{\alpha}9\xrightarrow{\alpha}10$, the transitions $3\xrightarrow{\alpha}4$ and $7\xrightarrow{\beta}8$ are missed, and hence, the diagnoser can only observe

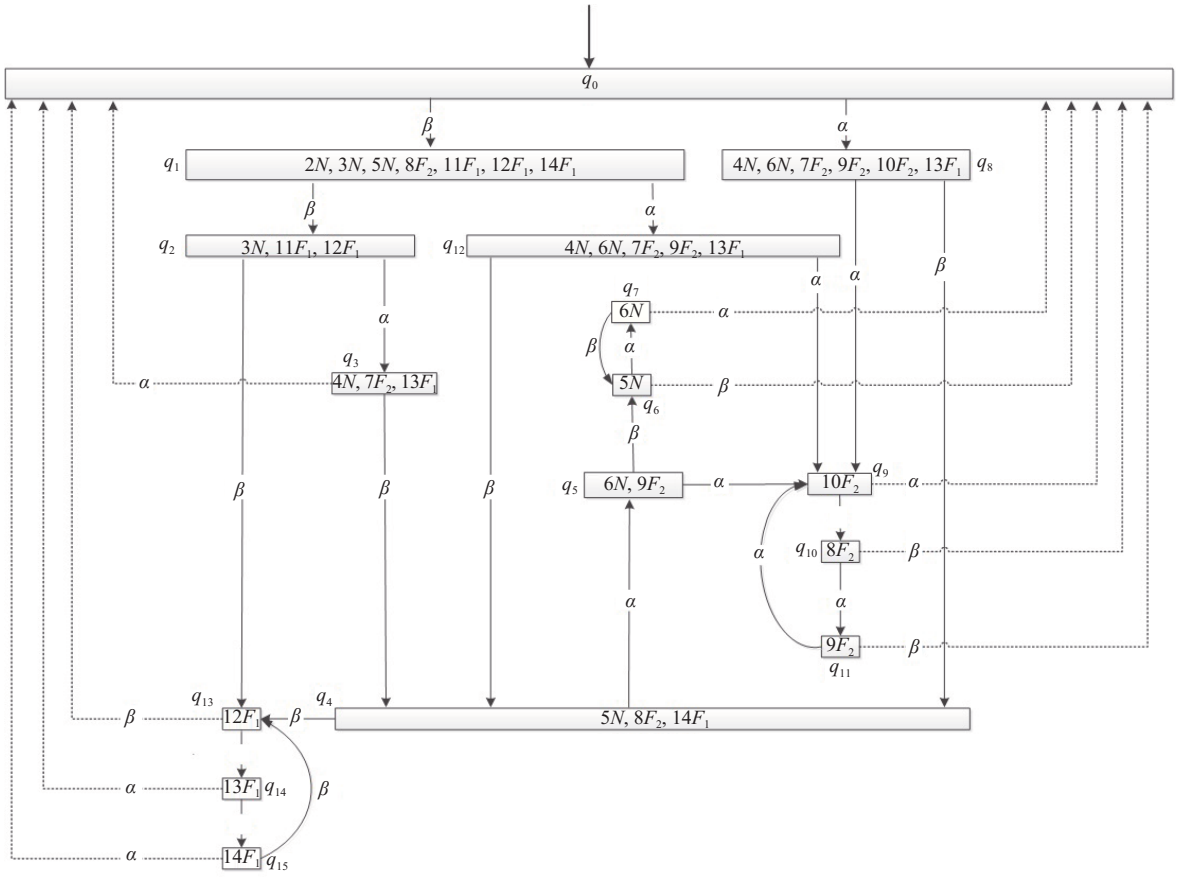

Fig. 4. The constructed diagnoser for the plant G_1 given in Fig. 3. The solid arrows represent δ_{dp} and the dashed arrows show the reset transitions, δ_{dr}.

$\beta\beta\alpha\alpha\beta\alpha\alpha$ instead of $\beta\beta\alpha\beta\alpha\alpha\beta\alpha\alpha$. Therefore, observing $\beta\beta\alpha\alpha\beta\alpha\alpha$, the diagnoser goes through the following sequence: $q_0 \xrightarrow{\beta} q_1 \xrightarrow{\beta} q_2 \xrightarrow{\alpha} q_3 \xrightarrow{\alpha} q_0 \xrightarrow{\beta} q_1 \xrightarrow{\alpha} q_{12} \xrightarrow{\alpha} q_9$. As it can be seen, after executing $q_0 \xrightarrow{\beta} q_1 \xrightarrow{\beta} q_2 \xrightarrow{\alpha} q_3$, the diagnoser realizes that an event is missed and will reset to q_0. Then, the diagnoser continues the diagnosis process and eventually can synchronize itself and will detect the fault by reaching q_9, which is an F_2-certain state, concluding that the fault f_2 has occurred in the system. Fig. 5 shows the executed runs in the plant and the diagnoser, under perfect and imperfect observations.

IV. Properties of the Constructed Diagnoser

Next, we outline some properties of the developed diagnoser and provide several lemmas, which will be used in future derivations.

Lemma 1: The constructed diagnoser is a deterministic automaton.

Proof: By construction the diagnoser's transition relation, δ_{di}, is deterministic. This can be observed in (3) and (4), in which for each state q and each event e, we search for all $(x,\ell) \in q$ and for all unobservable extensions of e, $t \in UE(e,x)$, and will find $(y, \nabla(\ell,t))$, where $y \in \delta(x,t)$. We aggregate all these pairs $(y, \nabla(\ell,t))$ as a new diagnoser state q'. Since all outgoing transitions from q by the event e and its unobservable extensions are already included in q', q' is the only member of $\delta_{dp}(q,e)$. A similar argument can be made for δ_{dr}. Therefore, $\delta_{di} = \delta_{dp} \cup \delta_{dr}$ is deterministic. ■

Lemma 2: For any $q \xrightarrow{\sigma} q'$ in G_{dp}, there exist $(x,\ell) \in q$, $(x',\ell') \in q'$, and a transition $x \xrightarrow{t} x'$ in G such that $P(t) = \sigma$.

Proof: This can be verified by the construction procedure in Step 2 of Algorithm 1. ■

Lemma 3: The extension closure of the observable behavior of the system under diagnosis, G, and the perfect asynchronous diagnoser, G_{dp}, are language equivalent, i.e., $ext(P(L(G))) = L(G_{dp})$.

Proof: Consider a string $t \in ext(P(L(G)))$, where $t = \sigma_1\sigma_2 \cdots \sigma_n$, $\sigma_i \in \Sigma_o$. There exist at least two strings $w = w_0e_1w_1e_2w_2 \cdots e_mw_m$ and $u = \sigma_1u_1\sigma_2u_2 \cdots \sigma_nu_n$, such that $w.u \in L(G)$, $u_i, w_j \in \Sigma_u^*$, and $\sigma_i, e_j \in \Sigma_o$ for all $i = 1,\ldots,n$, $j = 1,\ldots,m$. Correspondingly, there exist the states $z_1,\ldots,z_m \in X$ and $x_1,\ldots,x_n \in X$, such that $x_0 \xrightarrow{w_0} z_1 \xrightarrow{e_1w_1} z_2 \xrightarrow{e_2w_2} \cdots \xrightarrow{e_mw_m} z_m \xrightarrow{\sigma_1u_1} x_1 \xrightarrow{\sigma_2u_2} x_2 \cdots \xrightarrow{\sigma_nu_n} x_n$. According to Step 1 of Algorithm 1, $(x_0,\ell_0 = \{N\})$, $(z_1,\ell_1 = \nabla(\ell_0,w_0))$, $(z_2,\ell_2 = \nabla(\ell_1,e_1w_1))$, and $\ldots(z_m,\ell_m = \nabla(\ell_{m-1},e_mw_m))$ all are in q_0. Then, following Step 2 of Algorithm 1, for $(x_1,\bar{\ell}_1 = \nabla(\ell_m,\sigma_1u_1)) = \bar{\delta}_d((z_m,\ell_m), \sigma_1)$, there exists a state $q_1 \in Q_d$ such that $q_1 = \delta_{dp}(q_0,\sigma_1)$. Also, for all $(x_i,\bar{\ell}_i = \nabla(\ell_{i-1},\sigma_iu_i)) = \bar{\delta}_d((x_{i-1},\bar{\ell}_{i-1}),\sigma_i)$, there exists states $q_{i-1}, q_i \in Q_d$ such that $q_i = \delta_{dp}(q_{i-1},\sigma_i)$, forming the sequence $q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_n} q_n$, concluding that $t = \sigma_1\sigma_2 \cdots \sigma_n \in L(G_{dp})$.

On the other hand, consider a sequence $q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_n} q_n$, where $\sigma_1\sigma_2 \cdots \sigma_n \in L(G_{dp})$. Based on Lemma 2, there exists $x_1, x_2, x_3, \ldots, x_{n+1} \in X$ and strings $t_1, t_2, \ldots, t_n$ such that $P(t_1) = \sigma_1$, $P(t_2) = \sigma_2, \ldots, P(t_n) = \sigma_n$, and $x_1 \xrightarrow{t_1} x_2 \xrightarrow{t_2} x_3 \cdots \xrightarrow{t_n}$

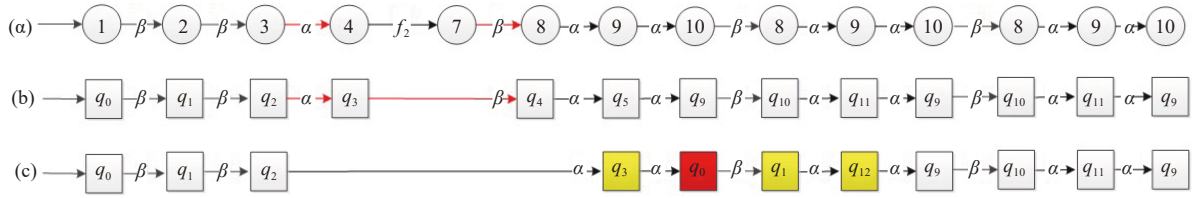

Fig. 5. (a) The executed run by the plant G_1; (b) the corresponding executed runs by the diagnoser in Fig. 4 under perfect observations; (c) the executed runs by the diagnoser in Fig. 4 under imperfect observations. Under imperfect observation,the diagnoser automatically rests to q_0 after missing the detectable observations and after few observations re-synchronizes itself with the plant executions.

x_{n+1}. Since, all strings in G are reachable from x_0, there exists a string s such that $x_0 \xrightarrow{s} x_1$, meaning that $P(st_1t_2\cdots t_n) = P(s)\sigma_1\sigma_2\cdots\sigma_n \in P(L(G))$, concluding that $\sigma_1\sigma_2\cdots\sigma_n \in ext(P(L(G)))$.

As proven above, any string in $ext(P(L(G)))$ is in $L(G_{dp})$ and vice versa, resulting in $ext(P(L(G))) = L(G_{dp})$ ■

V. Asynchronous Diagnosability

In Example 1, for a particular sequence of events, the diagnoser was able to detect the occurred fault in G_1 despite missing an observation. The question is, can the diagnoser always detect all faults in the case of any missed observation? If we look at the diagnosis mechanism, the proposed diagnoser detects a missed observation and resets to q_0, and then, resumes the diagnosis process. So, the previous question can be broken into two questions: 1) can the diagnoser always detect the missed observations? 2) can the diagnoser always diagnose a fault after arbitrary resetting to the initial state q_0? Next sections address these two questions. Before that, for the derivations in the next sections, the following definitions are needed.

Definition 2: A cycle in the diagnoser is called F_i-certain if all of its states are F_i-certain; otherwise, it is called a non-F_i-certain cycle.

Definition 3: A cycle of F_i-uncertain states in the diagnoser is called an F_i-uncertain cycle.

Definition 4 (F_i-indeterminate Cycle): A set of F_i-uncertain states $q_1, q_2, \ldots, q_n \in Q_d$ forms an F_i-indeterminate cycle if and only if

1) The states $q_1, q_2, \ldots, q_n$ form a cycle in the diagnoser, i.e., $\delta_d(q_k, e_k) = q_{k+1}$, for $k = 1,\ldots,n-1$, $\delta_d(q_n, e_n) = q_1$, and $e_k \in \Sigma_o$, for $k = 1,\ldots,n$.

2) The cycle $q_1, q_2, \ldots, q_n$ in the diagnoser can be inversely projected back to at least one cycle of non-F_i-faulty states and one cycle of F_i-faulty states in the original system G, i.e., each state of the cycle, q_k, contains (x_k, ℓ_k) and (x'_k, ℓ'_k) so that

a) $F_i \notin \ell_k$ and $F_i \in \ell'_k$, for all $(x_k, \ell_k) \in q_k$, $(x'_k, \ell'_k) \in q_k$, and $k = 1,\ldots,n$.

b) $x_1, x_2, \ldots, x_n$ form a cycle in G so that $x_{k+1} \in \delta(x_k, t_k)$, $k = 1,\ldots,n-1$, and $x_1 \in \delta(x_n, t_n)$, where $P(t_k) \in e_k$ for $k = 1, \ldots, n$.

c) $x'_1, x'_2, \ldots, x'_n$ form a cycle in G so that $x'_{k+1} \in \delta(x'_k, t'_k)$, $k = 1,\ldots,n-1$, and $x'_1 \in \delta(x'_n, t'_n)$, where $P(t'_k) \in e_k$ for $k = 1, \ldots, n$.

A. Asynchronous Diagnosability Under Perfect Observation

An observation may be missed anytime during the diagnosis process. If the diagnoser detects the missed observation, it resets to initial state q_0, missing the past history of observations (as they no longer are valid due to a missed observation). So, the diagnoser should be able to diagnose the faults solely based on the post-reset/activation observations. Since the reset can happen anytime, a fault should be distinguishable based on all sufficiently large sequences of events observed after a fault as stated in the following definition.

Definition 5 (Asynchronous Diagnosability Under Perfect Observations)[53]: The DES system G with the live language $L(G)$, is said to be F_i-asynchronously diagnosable with respect to the fault type F_i and the natural projection P, if for all $s \in L(G)$, $f \in \Sigma_{fi}$, $f \in s$, there exists an upper bound $n_i \in \mathbb{N}$, such that for any string $t \in L(G)/s$ with $|t| \geq n_i$, the following condition holds:

$$\{\forall uv \in L(G), P(v) = P(t)\} \Rightarrow f \in uv, f \in \Sigma_{fi}. \tag{5}$$

The system G is asynchronously diagnosable if it is F_i-asynchronously diagnosable with respect to all fault types F_i, $i = 1,...,m$.

Although Definition 5 describes the asynchronous diagnosability, it is very difficult to check the diagnosability condition given in (5) over all faulty strings in $L(G)$. Theorem 4, therefore, will provide the necessary and sufficient conditions to indirectly check the asynchronous diagnosability based on the structure of the diagnoser:

Lemma 4 (Asynchronous Diagnosability Theorem Under Perfect Observations)[51]: The plant G with the live language $L(G)$, and with the asynchronous diagnoser G_{dp}, is F_i-asynchronously diagnosable under perfect observations if and only if, there does not exist an F_i-indeterminate cycle in G_{dp}, i.e., there is no cycle of F_i-uncertain states in G_{dp} that can be projected back to a cycle of normal and F_i-faulty states in G.

Remark 1: Whether or not F_i-asynchronously diagnosable, we can construct the diagnoser G_d for the plant G. If the plant G is F_i-asynchronously diagnosable, then the constructed diagnoser can determine if a fault $f \in \Sigma_{fi}$ has occurred or not in a finite number of transitions. Therefore, it is preferred to have the plant G F_i-asynchronously diagnosable. But if the plant G is not F_i-asynchronously diagnosable, still the diagnoser G_d can be constructed and can provide its best estimation of the failures' status in G, though in some cases there might be an ambiguity in the occurrence of failures of type F_i, and the diagnoser cannot resolve the ambiguity even after observing a large number of transitions in G.

B. Asynchronous Diagnosability Under Imperfect Observation

If the diagnoser misses an observation, it should be able to

detect the missed observation based on the sequence of events observed after the diagnoser's last reset or activation. Since the diagnoser may have been activated or reset anytime asynchronous with the system under diagnosis, the missed observation should be distinguishable based on its precedent events, as stated below.

Definition 6 (Asynchronous Detectability of a Missed Observation): Consider the DES system G with the live language $L(G)$. The miss of observation of the event α, $\alpha \in \Sigma_o$, when the system G transits from a state $\tilde{x}$ to $\tilde{x}'$, $\tilde{x} \xrightarrow{\alpha} \tilde{x}'$, is said to be asynchronously detectable with respect to the natural projection P, if for any string $s \in L(G)$ that passes through $\tilde{x} \xrightarrow{\alpha} \tilde{x}'$, there exists an upper bound $n_i \in \mathbb{N}$, such that for any string $t \in L(G)/s$ with $|t| \geq n_i$, the following condition holds:

$$\{\forall uv \in L(G), P(v) = P(t)\} \Rightarrow uv \text{ passes through } \tilde{x}_1 \xrightarrow{\alpha} \tilde{x}'. \quad (6)$$

This definition indeed requires that after a certain number of observations, any string t that occurs after a missed observation be distinguishable in an asynchronous setting, i.e., not to be confused with another string in $ext(L(G))$. If we treat the missed observation as an unobservable faulty event f_m, then the detection of the missed observation is exactly as the same as asynchronously diagnosing f_m, when we should diagnose f_m solely based on post-fault occurrence information. Therefore, instead of directly checking the detectability of the transition $\tilde{x} \xrightarrow{\alpha} \tilde{x}'$, one can replace α in this transition with f_m and check for its asynchronous diagnosability. Note that with this practice, $\Sigma_{f_m} = \{f_m\}$ has the single member, f_m. Applying Definition 5 to check whether f_m is asynchronously diagnosable requires that for all $s \in L(G)$, $f_m \in s$, there should exist an upper bound $n_i \in \mathbb{N}$, such that for any string $t \in L(G)/s$ with $|t| \geq n_i$, the following condition holds:

$$\{\forall uv \in L(G), P(v) = P(t)\} \Rightarrow f_m \in uv, f_m \in \Sigma_{f_m}. \quad (7)$$

Example 2: Consider the manufacturing system in Example 1 whose model is captured by automaton G_1 in Fig. 3 with $\Sigma_o = \{\alpha,\beta\}$ and $\Sigma_u = \Sigma_f = \{f_1, f_2\}$. Missing the transitions $3 \xrightarrow{\alpha} 4$ and $7 \xrightarrow{\alpha} 8$ are asynchronously detectable. According to Definition 6, consider $s = \beta\beta\alpha \in L(G)$ which passes through $3 \xrightarrow{\beta} 4$, where $L(G)/s = f_2\beta(\alpha\alpha\beta)^* + \beta(\alpha\beta)^*$. Consider $u = \beta\beta$ and $v = f_1\beta\alpha\beta$, which might be confused by $t = \beta\alpha\beta$, for which $P(v) = P(t)$, but $f_m \notin uv$. However, if we wait for more observation, then $v = f_1\beta\alpha\beta\beta$, for which there is no $t \in L(G)/s$ such that $P(v) = P(t)$ to create confusion. This can be tested for all other strings in $L(G)/s \subset ext(L(G))$, which in general, it is not feasible to be tested for all strings. Alternatively, as shown in Fig. 6, we replace the events α in $3 \xrightarrow{\alpha} 4$ and β in $7 \xrightarrow{\beta} 8$ with f_m, and let $\Sigma_f = \{f_1, f_2, f_m\}$ with $\Sigma_{f_1} = \{f_1\}$, $\Sigma_{f_2} = \{f_2\}$, $\Sigma_{f_m} = \{f_m\}$. Then, we can construct an asynchronous diagnoser as shown in Fig. 7, which has two cycle of F_m-certain states, shown on orange, but has no cycle of F_m-uncertain states. Hence, f_m is asynchronously diagnosable, concluding that missing the transitions $3 \xrightarrow{\alpha} 4$ and $7 \xrightarrow{\alpha} 8$ are asynchronously detectable.

Same procedure can be applied when there are multiple missed transitions by replacing them with f_m, and check for the diagnosability of f_m. Next lemma explains that if a missed

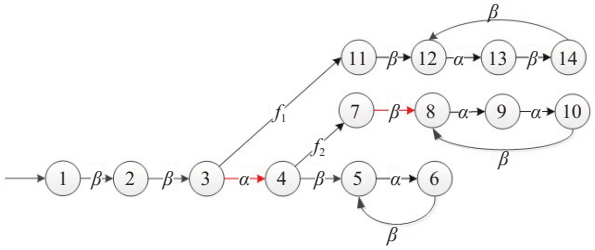

Fig. 6. To check if missing the transitions $3 \xrightarrow{\alpha} 4$ and $7 \xrightarrow{\beta} 8$ in G_1 in Fig. 3 are detectable, we replace them with f_m to check their asynchronous detectability. Int his way, $\Sigma_o = \{\alpha,\beta\}$ and $\Sigma_u = \Sigma_f = \{f_1, f_2, f_m\}$, with $\Sigma_{f_1} = \{f_1\}$, $\Sigma_{f_2} = \{f_2\}$, and $\Sigma_{f_m} = \{f_m\}$.

observation is detectable, the developed diagnoser G_{di} resets after a finite number of observations.

Lemma 5: Assume that the miss of event α in the transition $\tilde{x} \xrightarrow{\alpha} \tilde{x}'$ is asynchronously detectable. Upon the activation/reset of the diagnoser G_{di}, if the event α in the transition $\tilde{x} \xrightarrow{\alpha} \tilde{x}'$ is missed, G_{di} will reset after a finite number of transitions.

Proof: Consider that the plant G has executed the string s since the activation/reset of the diagnoser, and s has brought G to $\tilde{x}$, but the diagnoser G_{di} has missed the observation $\tilde{x} \xrightarrow{\alpha} \tilde{x}'$. This means that there exist s_1, s_2, $\tilde{x}''$, and $\hat{x}$ such that $s = s_1.\alpha.s_2$ and $\hat{x} \xrightarrow{s_1} \tilde{x} \xrightarrow{\alpha} \tilde{x}' \xrightarrow{s_2} \tilde{x}''$. However, due to missing the observation $\tilde{x} \xrightarrow{\alpha} \tilde{x}'$, instead of $P(s = s_1.\alpha.s_2)$, the diagnoser has observed $P(u = s_1.s_2)$. Then, the plant G continues execution of events which results in strings in $ext(P(L(G)))$. There is a finite number n, so that for a string t_1, generated by G with $P(t_1) = e_1e_2...e_n$, and its precedent string t_2 with $P(t_2) = e_{n+1}$, we have $P(s.t_1) \in ext(P(L(G)))$, but $P(s.t_1.t_2) \notin ext(P(L(G)))$. Otherwise, for an infinite size string $v = t_1.t_2 \in L(G)/s \subseteq ext(P(L(G)))$, $uv = s_1.s_2.t_1.t_2$ does not pass through $\tilde{x} \xrightarrow{\alpha} \tilde{x}'$, violating condition (6) and contradicting the fact that α is detectable. Since for all $t \in pr(t_1)$, $P(s.t) \in ext(P(L(G)))$, and since based on Lemma 3 $ext(P(L(G))) = L(G_{dp})$, for all $t \in pr(t_1)$, we have $P(s.t) \in L(G_{dp})$. In particular, $P(s.t_1) \in L(G_{dp})$, bringing the diagnoser to a state $q_1 = \delta_{dp}(q_0, P(s.t_1))$. Since the diagnoser is deterministic and $P(s.t_1.t_2) = P(s.t_1).e_{n+1} \notin ext(P(L(G))) = L(G_{dp})$, then $\delta_{dp}(q_0, P(s.t_1.t_2)) = \delta_{dp}(q_1, e_{n+1})$ is not defined. Based on Algorithm 1, $\delta_{di}(q_1, e_{n+1}) = \delta_{dr}(q_1, e_{n+1}) = q_0$, resetting G_{di}. ■

Putting all together, next theorems derive the conditions for diagnosing faults under imperfect observations.

Theorem 1 (Asynchronous Diagnosability Theorem Under Imperfect Observation): The plant G with the live language $L(G)$, and with the asynchronous diagnoser G_{di}, is F_i-asynchronously diagnosable under imperfect observation if there does not exist an F_i-indeterminate cycle in G_{di} and the missed observations are asynchronously detectable.

Proof: Since the transitions δ_{dr} corresponds to no transition in G, there will be no additional F_i-indeterminate cycle in G_{di}, i.e., the F_i-indeterminate cycles in G_{dp} and G_{di} are the same. Therefore, if there is no missed observation, the plant G is F_i-asynchronously diagnosable if there is no F_i-indeterminate cycle in G_{di}. However, if there are missed observations and if missed observations are detectable, after a finite number of observations, the diagnoser will reset as proven in Lemma 5,

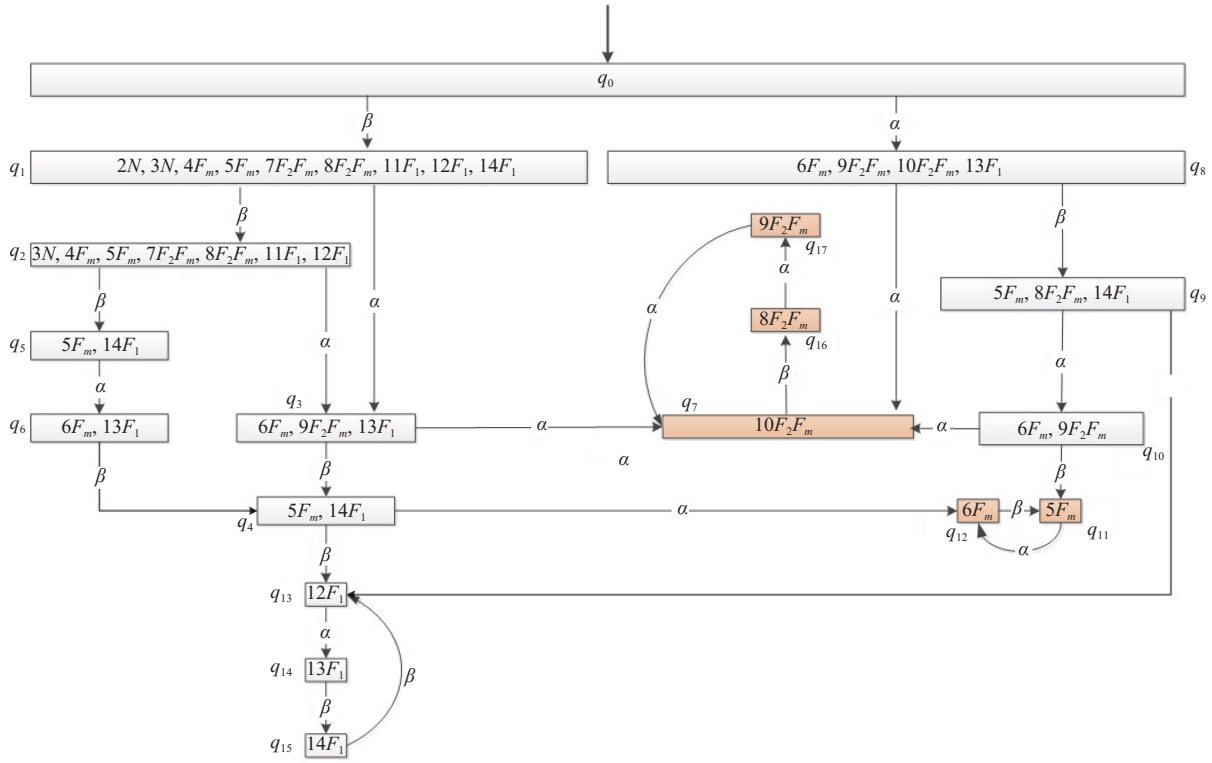

Fig. 7. The asynchronous diagnoser for the modified G_1 in Fig. 3, when replacing $3 \xrightarrow{\alpha} 4$ and $7 \xrightarrow{\beta} 8$ with f_m. The constructed diagnoser has two cycles of F_m-uncertain states, shown in orange, but has no F_m-indeterminate cycle. Hence, f_m is asynchronously diagnosable, concluding that missing the transitions $3 \xrightarrow{\alpha} 4$ and $7 \xrightarrow{\alpha} 8$ are asynchronously detectable.

and then will resume the diagnosis process. Since the plant is live and asynchronously diagnosable, after resetting, the asynchronous diagnoser G_{di} can diagnose the occurred faults, even if they have occurred before resetting the diagnoser. ■

Example 3: The diagnoser G_{di} for the plan G_1, shown in Fig. 4, has no cycle of uncertain states. Also, as shown in Example 2, the transitions $3 \xrightarrow{\alpha} 4$ and $7 \xrightarrow{\alpha} 8$ are asynchronously detectable. Therefore, according to Theorem 1, the plant G_1 is asynchronously diagnosable if missing the transitions $3 \xrightarrow{\alpha} 4$ and $7 \xrightarrow{\beta} 8$. For instance, as it is shown in Example 1, the fault f_2 is detected despite missing the transitions $3 \xrightarrow{\alpha} 4$ and $7 \xrightarrow{\beta} 8$.

VI. Conclusion

This paper developed a diagnosis technique which is capable of diagnosing faults under imperfect observations. A new concept of asynchronous detectability was introduced, which, if holds, allows to detect a miss observation from its post observations. Upon detecting a missed observation, the diagnoser resets and resumes the diagnosis process. It was proven that if the missed observations are asynchronously detectable and if the faults are asynchronously diagnosable, the developed diagnoser can detect the occurred fault despite missing the observations of asynchronously detectable events.

Acknowledgment

The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF, AFRL, OSD or the U.S. Government.

References

[1] A. Schumacher, T. Nemeth, and W. Sihn, "Roadmapping towards industrial digitalization based on an industry 4.0 maturity model for manufacturing enterprises," *Procedia CIRP*, vol. 79, pp. 409–414, 2019.

[2] E. A. Lee, "Cyber physical systems: Design challenges," in *Proc. 11th IEEE Int. Symp. Object and Component-Oriented Real-Time Distributed Computing*, Orlando, USA, 2008, pp. 363–369.

[3] R. Baheti and H. Gill, "Cyber-physical systems," in *The Impact of Control Technology*, T. Samad and A. M. Annaswamy, Eds. New York, USA: IEEE Control Systems Society, 2011, pp. 161–166.

[4] D. O. M. Sanchez, "Sustainable development challenges and risks of industry 4.0: A literature review," in *Proc. Global IoT Summit*, Aarhus, Denmark, 2019, pp. 1–6.

[5] M. M. Alani and M. Alloghani, "Security challenges in the industry 4.0 era," in *Industry 4.0 and Engineering for a Sustainable Future*, M. Dastbaz and P. Cochrane, Eds. Cham, Germany: Springer, 2019, pp. 117–136.

[6] V. Alcácer and V. Cruz-Machado, "Scanning the industry 4.0: A literature review on technologies for manufacturing systems," *Eng. Sci. Technol., Int. J.*, vol. 22, no. 3, pp. 899–919, Jun. 2019.

[7] S. Jeschke, C. Brecher, T. Meisen, D. Özdemir, and T. Eschert, "Industrial internet of things and cyber manufacturing systems," in *Industrial Internet of Things*, S. Jeschke, C. Brecher, H. B. Song, and D. B. Rawat, Eds. Cham, Germany: Springer, 2017, pp. 3–19.

[8] R. M. Murray, K. J. Astrom, S. P. Boyd, R. W. Brockett, and G. Stein, "Future directions in control in an information-rich world," *IEEE Control Syst. Mag.*, vol. 23, no. 2, pp. 20–33, Apr. 2003.

[9] A. D. Pouliezos and G. S. Stavrakakis, *Real Time Fault Monitoring of Industrial Processes*. Dordrecht, Netherlands: Springer, 1994.

[10] P. M. Frank, "Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: A survey and some new results," *Automatica*, vol. 26, no. 3, pp. 459–474, May 1990.

[11] A. Diez-Olivan, J. Del Ser, D. Galar, and B. Sierra, "Data fusion and machine learning for industrial prognosis: Trends and perspectives towards industry 4.0," *Inf. Fusion*, vol. 50, pp. 92–111, Oct. 2019.

[12] J. Chen and R. J. Patton, *Robust Model-Based Fault Diagnosis for Dynamic Systems*. Boston, USA: Springer, 1999.

[13] V. Venkatasubramanian, R. Rengaswamy, K. W. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis: Part I: Quantitative model-based methods," *Comput. Chem. Eng.*, vol. 27, no. 3, pp. 293–311, Mar. 2003.

[14] A. Bhagwat, R. Srinivasan, and P. R. Krishnaswamy, "Fault detection during process transitions: A model-based approach," *Chem. Eng. Sci.*, vol. 58, no. 2, pp. 309–325, Jan. 2003.

[15] R. Isermann, "Model-based fault-detection and diagnosis - status and applications," *Annu. Rev. Control*, vol. 29, no. 1, pp. 71–85, 2005.

[16] S. Rajakarunakaran, P. Venkumar, D. Devaraj, and K. S. P. Rao, "Artificial neural network approach for fault detection in rotary system," *Appl. Soft Comput.*, vol. 8, no. 1, pp. 740–748, Jan. 2008.

[17] Y. F. Zhou, J. Hahn, and M. S. Mannan, "Fault detection and classification in chemical processes based on neural networks with feature extraction," *ISA Trans.*, vol. 42, no. 4, pp. 651–664, Oct. 2003.

[18] Y. M. Zhang and J. Jiang, "Issues on integration of fault diagnosis and reconfigurable control in active fault-tolerant control systems," in *Fault Detection, Supervision and Safety of Technical Processes 2006*, H. Y. Zhang, Ed. Oxford, UK: Elsevier Science Ltd, 2007, pp. 1437–1448.

[19] X. D. Zhang, T. Parisini, and M. M. Polycarpou, "Adaptive fault-tolerant control of nonlinear uncertain systems: An information-based diagnostic approach," *IEEE Trans. Autom. Control*, vol. 49, no. 8, pp. 1259–1274, Aug. 2004.

[20] Y. Zheng, H. J. Fang, and H. O. Wang, "Takagi-sugeno fuzzy-model-based fault detection for networked control systems with Markov delays," *IEEE Trans. Syst.*, *Man*, *Cybern.*, *Part B Cybern.*, vol. 36, no. 4, pp. 924–929, Aug. 2006.

[21] P. Y. Zhang, S. Shu, and M. C. Zhou, "An online fault detection model and strategies based on SVM-grid in clouds," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 2, pp. 445–456, Mar. 2018.

[22] W. S. Lee, D. L. Grosh, F. A. Tillman, and C. H. Lie, "Fault tree analysis, methods, and applications - a review," *IEEE Trans. Reliab.*, vol. R-34, no. 3, pp. 194–203, Aug. 1985.

[23] J. D. Andrews and S. J. Dunnett, "Event-tree analysis using binary decision diagrams," *IEEE Trans. Reliab.*, vol. 49, no. 2, pp. 230–238, Jun. 2000.

[24] D. N. Pandalai and L. E. Holloway, "Template languages for fault monitoring of timed discrete event processes," *IEEE Trans. Autom. Control*, vol. 45, no. 5, pp. 868–882, May 2000.

[25] S. R. Das and L. E. Holloway, "Characterizing a confidence space for discrete event timings for fault monitoring using discrete sensing and actuation signals," *IEEE Trans. Syst.*, *Man*, *Cybern. - Part A*: *Syst. Hum.*, vol. 30, no. 1, pp. 52–66, Jan. 2000.

[26] M. Gromov and T. A. C. Willemse, "Testing and model-checking techniques for diagnosis," in *Testing of Software and Communicating Systems*, A. Petrenko, M. Veanes, J. Tretmans, and W. Grieskamp, Eds. Berlin, Heidelberg, Germany: Springer, 2007, pp. 138–154.

[27] F. Cicirelli, A. Furfaro, and L. Nigro, "Model checking time-dependent system specifications using time stream petri nets and uppaal," *Appl. Math. Comput.*, vol. 218, no. 16, pp. 8160–8186, Apr. 2012.

[28] U. Lerner, R. Parr, D. Koller, G. Biswas, "Bayesian fault detection and diagnosis in dynamic systems," in *Proc. 17th Nat. Conf. Artificial Intelligence and Twelfth Conf. Innovative Applications of Artificial Intelligence*, Austin, USA, 2000, pp. 531–537.

[29] Z. Simeu-Abazi, M. D. Mascolo, and M. Knotek, "Fault diagnosis for discrete event systems: Modelling and verification," *Reliab. Eng. Syst. Saf.*, vol. 95, no. 4, pp. 369–378, Apr. 2010.

[30] M. P. Cabasino, A. Giua, and C. Seatzu, "Fault detection for discrete event systems using petri nets with unobservable transitions," *Automatica*, vol. 46, no. 9, pp. 1531–1539, Sept. 2010.

[31] R. Ammour, E. Leclercq, E. Sanlaville, and D. Lefebvre, "Datation of faults for markovian stochastic dess," *IEEE Trans. Autom. Control*, vol. 64, no. 7, pp. 2961–2967, Jul. 2019.

[32] J. Lunze, "Discrete-event modelling and fault diagnosis of discretely controlled continuous systems," in *Analysis and Design of Hybrid Systems 2006*, C. Cassandras, A. Giua, C. Seatzu, and J. Zaytoon, Eds. Amsterdam, Netherlands: Elsevier, 2006, pp. 229–234.

[33] K. Schmidt, "Abstraction-based failure diagnosis for discrete event systems," *Syst. Control Lett.*, vol. 59, no. 1, pp. 42–47, Jan. 2010.

[34] G. G. Rigatos, "Fault detection and isolation based on fuzzy automata," *Inf. Sci.*, vol. 179, no. 12, pp. 1893–1902, May 2009.

[35] P. Philips, K. B. Ramkumar, K. W. Lim, H. A. Preisig, and M. Weiss, "Automaton-based fault detection and isolation," *Comput. Chem. Eng.*, vol. 23, no. Suppl, pp. S215–S218, Jun. 1999.

[36] G. P. Bhandari and R. Gupta, "Fault diagnosis in service-oriented computing through partially observed stochastic petri nets," *Serv. Oriented Comput. Appl.*, vol. 14, no. 1, pp. 35–47, Mar. 2020.

[37] F. Lin, "Diagnosability of discrete event systems and its applications," *Discrete Event Dyn. Syst.*, vol. 4, no. 2, pp. 197–212, May 1994.

[38] N. Ran, H. Y. Su, and S. G. Wang, "An improved approach to test diagnosability of bounded petri nets," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 2, pp. 297–303, Apr. 2017.

[39] R. Su and W. M. Wonham, "Global and local consistencies in distributed fault diagnosis for discrete-event systems," *IEEE Trans. Autom. Control*, vol. 50, no. 12, pp. 1923–1935, Dec. 2005.

[40] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. 2nd ed. New York, USA: Springer, 2008.

[41] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proc. IEEE*, vol. 77, no. 1, pp. 81–98, Jan. 1989.

[42] P. Baroni, G. Lamperti, P. Pogliano, and M. Zanella, "Diagnosis of a class of distributed discrete-event systems," *IEEE Trans. Syst.*, *Man*, *Cybern.-Part A*: *Syst. Hum.*, vol. 30, no. 6, pp. 731–752, Nov. 2000.

[43] A. Philippot, M. Sayed-Mouchaweh, and V. Carré-Ménétrier, "Unconditional decentralized structure for the fault diagnosis of discrete event systems," in *IFAC Proc. Volumes*, vol. 40, no. 6, pp. 67–72, 2007.

[44] S. Bhattacharyya, R. Kumar, and Z. Huang, "A discrete event systems approach to network fault management: Detection and diagnosis of faults," *Asian J. Control*, vol. 13, no. 4, pp. 471–479, Jul. 2011.

[45] S. Bhattacharyya, Z. Huang, V. Chandra, and R. Kumar, "A discrete event systems approach to network fault management: Detection & diagnosis of faults," in *Proc. American Control Conf.*, Boston, USA, 2004, pp. 5108–5113.

[46] M. Agarwal, S. Purwar, S. Biswas, and S. Nandi, "Intrusion detection system for PS-Poll DoS attack in 802.11 networks using real time discrete event system," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 4, pp. 792–808, 2017.

[47] A. White and A. Karimoddini, "Event-based diagnosis of flight maneuvers of a fixed-wing aircraft," *Reliab. Eng. Syst. Saf.*, vol. 193, pp. 106609, Jan. 2020.

[48] 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, Sept. 1995.

[49] S. Hashtrudi Zad, R. H. Kwong, and W. M. Wonham, "Fault diagnosis in timed discrete-event systems," in *Proc. 38th IEEE Conf. Decision and Control*, Phoenix, USA, 1999, pp. 1756–1761.

[50] M. M. Karimi, A. Karimoddini, A. P. White, and I. W. Bates, "Event-based fault diagnosis for an unknown plant," in *Proc. IEEE 55th Conf. Decision and Control*, Las Vegas, USA, 2016, pp. 7216–7221.

[51] A. White and A. Karimoddini, "Asynchronous fault diagnosis of discrete event systems," in *Proc. American Control Conf.*, Seattle, USA, 2017, pp. 3224–3229.

[52] A. White and A. Karimoddini, "Semi-asynchronous fault diagnosis of discrete event systems," in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, Budapest, Hungary, 2016, pp. 3961–3966.

[53] A. White, A. Karimoddini, and R. Su, "Fault diagnosis of discrete event systems under unknown initial conditions," *IEEE Trans. Autom. Control*, vol. 64, no. 12, pp. 5246–5252, Dec. 2019.

[54] R. Fritz and P. Zhang, "Overview of fault-tolerant control methods for discrete event systems," *IFAC-PapersOnLine*, vol. 51, no. 24, pp. 88–95, 2018.

[55] J. Dai, A. Karimoddini, and H. Lin, "Achieving fault-tolerance and

safety of discrete-event systems through learning," in *Proc. American Control Conf.*, Boston, USA, 2016, pp. 4835–4840.

[56] M. Karimadini, A. Karimoddini, and A. Homaifar, "A survey on fault-tolerant supervisory control," in *Proc. IEEE 61st Int. Midwest Symp. Circuits and Systems*, Windsor, Canada, 2018, pp. 733–738.

[57] R. Su, "Distributed trace estimation under timing mismatch and channel distortion," *IEEE Trans. Autom. Control*, vol. 53, no. 10, pp. 2409–2414, Nov. 2008.

[58] J. C. Basilio and S. Lafortune, "Robust codiagnosability of discrete event systems," in *Proc. American Control Conf.*, St. Louis, USA, 2009, pp. 2202–2209.

[59] D. Thorsley, T. S. Yoo, and H. E. Garcia, "Diagnosability of stochastic discrete-event systems under unreliable observations," in *Proc. American Control Conf.*, Seattle, USA, 2008, pp. 1158–1165.

[60] L. K. Carvalho, J. C. Basilio, and M. V. Moreira, "Robust diagnosis of discrete event systems against intermittent loss of observations," *Automatica*, vol. 48, no. 9, pp. 2068–2078, Sept. 2012.

[61] L. K. Carvalho, M. V. Moreira, and J. C. Basilio, "Diagnosability of intermittent sensor faults in discrete event systems," *Automatica*, vol. 79, pp. 315–325, May 2017.

[62] S. T. S. Lima, J. C. Basilio, S. Lafortune, and M. V. Moreira, "Robust diagnosis of discrete-event systems subject to permanent sensor failures," *IFAC Proc. Volumes*, vol. 43, no. 12, pp. 90–97, 2010.

Alejandro White (M'20) is a Postdoctoral Fellow within the Mechanics Division of the Vehicle Technology Directorate of the United States Army Combat Capabilities Development Command Army Research Laboratory. He received the bachelor of science in applied mathematics in 2003, and a bachelor of science in electrical engineering in 2004, from North Carolina A&T State University, USA. He then received the master of science in electrical engineering with a concentration in controls, from Virginia Tech in 2007. After completing the master's degree, he served as a System's Engineer in the avionics industry. In 2018 he received the Ph.D. degree in electrical engineering from North Carolina A&T State University. His research interests include fault diagnostics of discrete event systems, flight control systems, and machine learning.

Ali Karimoddini (SM'15) is an Associate Professor in the ECE Department at North Carolina A&T State University, USA. He received the bachelor of electrical and electronics engineering from the Amirkabir University of Technology, Iran, in 2003. He then received the master of science in instrumentation and automation engineering from Petroleum University of Technology, Iran, in 2007. In 2008, he joined the National University of Singapore, Singapore, to pursue the Ph.D. degree, and then, he joined the University of Notre Dame, USA, to conduct the postdoctoral studies. His research interests include cyber-physical systems, control and robotics, resilient control systems, flight control systems, multi-agent systems, and human-machine interactions. He is the Director of NC-CAV Center of Excellence on Advanced Transportation, Director of ACCESS Laboratory, and the Deputy Director of TECHLAV DoD Center of Excellence on Autonomy. His research has been supported by different federal funding agencies and industrial partners. He is a Member of AIAA, ISA, and AHS.

Mohammad Karimadini received the bachelor of electrical and electronics engineering from Azad University of Tehran, Iran, in 1996. He then joined industry and had worked as an Engineer and Supervisor in the Instrumentations and Control Department of National Iranian Copper Industries Company (NICICO) from 1996 to 2004. He received the master of science in control and automation in 2006 and Ph.D. in control engineering in 2012, respectively, from UPM University of Malaysia and National University of Singapore (NUS), Singapore, followed by two postdoctoral fellowships at NUS. He is currently an Assistant Professor in Department of Electrical Engineering, Arak University of Technology (ARAKUT), Iran, and Vice-President for Technology and Innovation in Arak Science and Technology Park (ASTP). His research interests include supervisory control of discrete event systems, decentralized cooperative control of multi-agent systems, industrial automation, and data-driven knowledge discovery and decision making.