Prognosis of Patterns in Networked Discrete-Event Systems with Communication Delays

Rui Zhao, Fuchun Liu

School of Computers, Guangdong University of Technology, Guangzhou 510006 China (Tel: +8613160809171; e-mail: zhaorui118204@163.com, corresponding author: fliu2011@163.com)

Abstract: This paper investigates the problem of prognosis of patterns in the context of networked discrete-event systems with communication delays. In this problem, the pattern to be predicted describes event sequences modeled by a finite state automaton; communication delays are delays in the observation channel and the delays are random and bounded. Given a pattern, in order to check whether an observation-delay system is predictable with respect to the pattern and predict the pattern k steps before it occurs, a new notion of k-step pattern predictability is defined and an off-line algorithm is presented for verifying the property. For a predictable system, a predictor is constructed to predict the occurrence of a pattern on-line during the evolution of the system.

Keywords: discrete-event systems, predictability, pattern.

1. INTRODUCTION

Networked discrete-event systems (DESs) are able to fulfil autonomous and cooperative tasks by exchanging information over a shared communication network. So they have been applied to many practical systems, such as intrusion detection in networked systems, power distribution networks etc. In the networked DESs, communication delays and losses are unavoidable, particularly for communication over wireless networks. Recently, the impacts of communication delays and losses on networked DESs are better concerned in the field of supervisory control. Several methodologies have been developed for different types of control problems; see (Park, 2007; Liu, 2009; Lin, 2014; Shu, 2015; Zgorzelski, 2018) for details. This paper considers the problem of predicting sequence patterns from the behaviour of a partially observed networked DES with communication delays.

The main objective of failure prediction is to develop methodologies for predicting the occurrences of possible failures arising in the operation of a dynamic system. In the last decade, the research problem has received considerable attention (Genc, 2009; Chang, 2013; Yin, 2016, 2019; Benmessahel, 2017; Liu, 2019; Zhao, 2019). And most of prior works on failure prediction of DESs pertain to the prediction of single failure event. However, in many real systems, a failure may be caused by a sequence of several events, where any single of them is not a failure event. In the problem of prognosis, when the predicted failure is caused by a sequence of one or more observable or unobservable events, the method is called pattern prognosis.

Pattern prognosis is to predict the occurrence of a pattern that leads or causes a system to fail or malfunction so that the system operator can take preventative measures. In recent years, the problem of diagnosis and prognosis of a pattern has been studied by many works in the DES literatures. T. Jéron and H. Marchand et al. formally proposed the notion of supervision patterns in discrete event systems diagnosis (Jéron, 2006). S. Genc and S. Lafortune studied the problem of diagnosis of a pattern in a partially-observed DES and defined two types of pattern diagnosability: S-type and T-type (Genc, 2006). T. Jéron et al. investigated the problem of predicting the occurrence of a pattern in a DES and defined the new notion of predictability of patterns (Jéron, 2008). L. Ye et al. proposed the approaches for pattern diagnosis in distributed DESs (Ye, 2009, 2012). In 2017, G. Lamperti and X. Zhao considered diagnosis of active systems and proposed a method for patterns based on semantic (Lamperti, 2017). X. Geng et al. investigated pattern diagnosis in a stochastic DES and formalized the definition of PA-diagnosability and PAA diagnosability (Geng, 2020).

However, the above results of patterns are not adapted to networked DESs with communication delays. Due to the communication delays and the random nature of the delays, a system that is originally pattern predictable may become unpredictable and the designed predictor may not accurately predict whether a given pattern will occur. Therefore, it is necessary to concern on the problem not encountered in the general pattern prognosis.

The paper aims to deal with the new issue of how to predict the occurrence of a pattern in the networked DESs with communication delays. In our work, the system is modeled as a partially-observed networked DES with communication delays. The sequence pattern to be predicted is modeled by a finite state automaton (FSA). Given a pattern, our objective is two-fold: one is to detect whether a system with communication delays is pattern predictable; the other is to design a predictor to predict the occurrences of patterns as soon as possible.

Our main contributions are: 1) formal definition of the notion of k-step pattern predictability. This notion is a generalization

of the previous definition of predictability for patterns in (Jéron, 2008); 2) derivation of the necessary and sufficient condition for verifying k-step pattern predictability; 3) and an algorithm of constructing a predictor for detecting on-line whether a pattern will occur after k steps.

The results presented in this paper are mainly related to the works of (Lin, 2014; Shu, 2015; Genc, 2006; Jéron, 2008). The early works of S. Shu and F. Lin (Lin, 2014; Shu, 2015) considered the problem of supervisor control of networked DESs. They did not address the pattern prognosis problem as the paper does. S. Genc and S. Lafortune discussed the diagnosis of a pattern in partially-observed DESs (Genc, 2006), which is different from the pattern prognosis in our work. There is a connection between the pattern prognosis and pattern diagnosis, but they are two different problems. Our work is closer to (Jéron, 2008), where the problem of prognosis of a pattern in a partially-observed DES was investigated and they assumed the system is live. However, this paper considers pattern prognosis in the context of the networked DESs with communication delays and relaxes the assumption.

The remainder of the paper is organized as follows. In section 2, the necessary background on DESs is presented. In section 3, the notion of k-step pattern predictability is proposed. In section 4, the necessary and sufficient condition of k-step pattern predictability is derived and an algorithm is presented for verifying the pattern predictability. In section 5, a predictor is designed to infer about the occurrence of a pattern on-line during the evolution of the system. Finally, in section 6, conclusions are drawn.

2. PRELIMINARIES

In the networked DESs, communication delays are inevitable. The communication delays in observation channel are called observation delays (Shu, 2015). This paper only considers the observation delays in the communication delays and the following assumptions are made in this work: 1) the delays are random but they are bounded; 2) the delays will not change the order of observation; 3) the delays are measured by the number of events that are executed by the plant after the occurrence of an event and before its observation.

Here, assume the observation delays are upper bounded by μ events. That is to say, when an event is observed from the system behavior, the system may have executed $\mu+1$ events at most.

A networked DES with observation delays μ can be denoted as

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

where X is the finite state space, Σ is the set of events, $\delta: X\times\Sigma\to X$ is the transition function, X_m is the set of marked states and $x_0\in X$ is the initial state of the system. Here, by default, all states are marked states. That is $X_m=X$. Σ^* denotes the set of all finite or infinite strings over Σ, including the empty string ϵ. The transition function δ is also extended to $\delta: X\times\Sigma^*\to X$ recursively by: for any $x\in X$, $s\in\Sigma^*$ and $\sigma\in\Sigma, \delta(x,\varepsilon)=x$ and $\delta(x,s\sigma)=\delta(\delta(x,s),\sigma)$. When $\mu=0$, the networked system reduces to a general DES, denoted by G.

The language generated by $\hat{G}$ is defined as $L=\{s\in\Sigma^*:\delta(x_0,s)\in X\}$ and it is often denoted by $L(\hat{G})$. $L_m(\hat{G})$ is a set of strings that can reach the marked states and $L_m(\hat{G})\subseteq L(\hat{G})$. Given a trace s originating from x_0, denote L/s as the post language of L after s, i.e., $L/s=\{t\in\Sigma^*: st\in L\}$ and denote $\overline{s}$ as the prefix language of s. Given a string s, the length of s (number of events including repetitions) is denoted by $\|s\|$.

All events of $\hat{G}$ are partitioned as $\Sigma=\Sigma_o\bigcup\Sigma_{uo}$, where Σ_o denotes the set of observable events and Σ_{uo} denotes the set of unobservable events. φ denotes empty sets.

Let $P:\Sigma^*\to\Sigma_o^*$ denote the usual projection operator which filters the unobservable events from a trace, and the inverse projection is $P^{-1}(y)=\{s\in L: P(s)=y\}$.

Due to the communication delays, the strings observed by the observer are often inconsistent with those generated by the system. Let s be a string and $\|s\|>\mu$, use $s_{\lrcorner\mu}$ to denote the string obtained by removing the last μ events from the string s. And the observation mapping with communication delays χ_μ: $\Sigma^*\to\Sigma_o^*$ is defined as $\chi_\mu(s)=\{t\in\Sigma_o^*: t=P(s_{\lrcorner(\mu-j)}), j=0,1,2,...,\mu\}$. Its inverse projection is $\chi_\mu^{-1}(t)=\{s\in L:\chi_\mu(s)=t\}$. Note that, if $\mu=0$, $\chi_\mu(s)$ is reduced to $P(s)$. And define $\chi_\mu(\varepsilon)=P(\varepsilon)=\varepsilon$.

Let x be a state of $\hat{G}$ (i.e. $x\in X$). The set of states reachable from x after an observable string s (i.e. $s\in\Sigma_o^*$) is denoted by

$$\Re(x,s)=\{x'\in X:\exists t\in\Sigma^* \text{ s.t. } \chi_\mu(t)=s\wedge\delta(x,t)=x'\}. \tag{2}$$

If $s=\varepsilon$, $\Re(x,\varepsilon)=\{x'\in X:\exists t\in\Sigma^* \text{ s.t. } P(t)=\varepsilon\wedge\delta(x,t)=x'\}$.

3. K-STEP PATTERN PREDICTABILITY

In this section, pattern predictability presented by T. Jéron et al. (Jéron, 2008) is reviewed first and an example is provided, then the notion of k-step pattern predictability is introduced. Finally, the relationship between the pattern predictability and k-step pattern predictability is analyzed.

Before presentation of pattern predictability, let's give some basic concepts first.

Definition 1 (substring and subsequence): Given two strings t and μ, if $\mu=stv$ and $s,v\in\Sigma^*$, then t is substring of μ, denoted as $t\Xi\mu$; if the string t is obtained by deleting zero or more events (either continuous or discontinuous) from μ, then t is called the subsequence of μ, denoted as $t\Upsilon\mu$.

Definition 2 (pattern and pattern string): Given event set Σ. A pattern can be seen as a finite set of bounded strings over Σ, denoted by Θ. An element of Θ is called a pattern string.

Definition 3 (a subset of L): Given language L and pattern Θ, a subset of L, denoted by $\wp \subseteq L$, is defined as the set of traces in L that take a pattern string of Θ as their substring or subsequence, i.e. $\wp = \{s \in L : \exists u \in \Theta \text{ s.t. } u \Xi s \text{ or } u \Upsilon s\}$.

Definition 4 (pattern trace): Given $\wp$. A subset of $\wp$, denoted by $\psi_{\wp}(\Theta)$, is defined as the set of strings in $\wp$ ending with an event which is the last event of a pattern string in Θ, i.e., $\Psi_{\wp}(\Theta) = \{s\sigma \in \wp : \exists u\sigma \in \Theta \text{ s.t. } u\sigma \Xi s\sigma \text{ or } u\sigma \Upsilon s\sigma\}$. An element of $\psi_{\wp}(\Theta)$ is called a pattern trace.

Definition5 (the minimum length of pattern traces): Given pattern trace set $\psi_{\wp}(\Theta)$. For any pattern trace $s \in \psi_{\wp}(\Theta)$, $l_{\min}(s)$ denotes the minimum length of pattern traces, i.e., $l_{\min}(s) = \min_{s \in \Psi_{\wp}(\Theta)} (\|s\|)$.

Formally, pattern predictability introduced in (Jéron, 2008) is defined as follows.

Definition 6 (Pattern Predictability):A prefix-closed language L generated by $\hat{G}$ is said to be pattern predictable w.r.t. Θ, if for $\forall s \in \psi_{\wp}(\Theta)$, there exist $n \in \mathbb{N}$ and $uv \in \overline{s}(uv \neq s)$ s.t. $[\mathbb{Z}(u) = 1]$,

where $\mathbb{N}$ is the set of natural numbers and the function $\mathbb{Z}(u)$ is defined as follows:

$$\mathbb{Z}(u) = \begin{cases} 1, \text{if } (\forall u' \in (P^{-1}P(u)) \cap L) \\ (\forall t \in L/u')(\|t\| \geq n \Rightarrow (u't \in \wp)) \\ 0, otherwise \end{cases} ; \quad (3)$$

The next example which comes from (Jéron, 2008) illustrates the notion.

Example 1: Consider the plant G_1 shown in Fig.1, in which the set of events is $\Sigma = \{f_1, f_2, a, b, c\}$. Let $\Sigma_{uo} = \{a, f_1, f_2\}$ be the set of unobservable events and event sequences $f_1 f_2$ and $f_2 f_1$ be the patterns to be predicted.

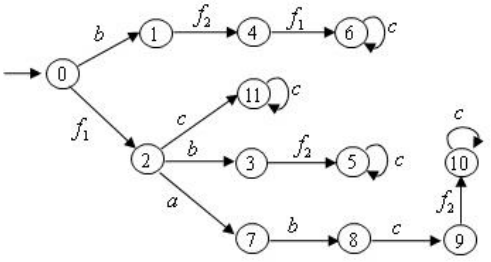

Fig. 1. Plant G_1.

As seen from Fig.1, there are three branches containing the patterns: $bf_2f_1c^*$, $f_1bf_2c^*$ and $f_1abcf_2c^*$. After the observation of b from the behavior of the system, it can be determined that the patterns will occur in the future. Therefore, G_1 is pattern predictable as described in (Jéron, 2008).

Assume that G_1 is a networked system and the maximum observation delay is one event (i.e. μ=1). When event b is observed, the system is either in state 1, 3 or 8, or in state 4, 5 or 9. If the system is in the state 5, it may be too late to predict the occurrence of the pattern $f_1 f_2$ since the event sequence $f_1 b f_2$ has been triggered. In the case, the pattern in the G_1 cannot be predicted before it occurs. In other words, the pattern predictability proposed by T. Jéron et al. does not describe the case.

Therefore, it is necessary to propose a new notion of pattern predictability in the case of observation delays.

Definition 7 (k-step Pattern Predictability): Let the maximum observation delays of $\hat{G}$ be μ. A prefix-closed language L generated by $\hat{G}$ is said to be k-step pattern predictable w.r.t. Θ and μ, if $(n \in \mathbb{N})$ s.t. $(\forall s \in \psi_{\wp}(\Theta))(\exists uv \in \overline{s} \wedge uv \neq s \wedge \|v\| \geq \mu + k)[\mathbb{C}(u) = 1]$,

where $\mathbb{N}$ is the set of natural numbers and the function $\mathbb{C}(u)$ is defined as follows:

$$\mathbb{C}(u) = \begin{cases} 1, \text{if } (\forall u' \in (\chi_{\mu}^{-1}\chi_{\mu}(u)) \cap L) \\ (\forall t \in L/u')(\|t\| \geq n \Rightarrow (u't \in \wp)) \\ 0, otherwise \end{cases} ; \quad (4)$$

Intuitively, the definition means that for any trace s containing a pattern string and ending with an event which is the last event of the pattern string, there exists a strict prefix uv and the length of v is greater than or equal to the sum of steps of maximum delays and advance prediction, such that any trace compatible with observation $\chi_{\mu}(u)$ will inevitably be extended into a string that contains a pattern string. That is, under the case of observation delays, upon the observation of $\chi_{\mu}(u)$, one can predict a pattern k steps before it occurs.

Proposition 1: Given μ and pattern Θ, if $\hat{G}$ is k-step pattern predictable, then $\hat{G}$ is $(k-i)-step$ pattern predictable for $i = 1, 2, ..., k-1$.

Proof: if $\hat{G}$ is k-step pattern predictable, then for all $s \in \Psi_{\wp}(\Theta)$, there exists a $uv \in \overline{s}$ and $uv \neq s$ such that $\|v\| \geq (\mu + k)$ and $\mathbb{C}(u) = 1$. For the uv, if $\|v\| \geq (\mu + k)$ then $\|v\| \succ (\mu + k - i)$. Therefore, $\mathbb{C}(u) = 1$ still holds. That is, $\hat{G}$ is $(k-i)-$ step pattern predictable.

Now, Let us relate pattern predictability (Jéron, 2008) to the notion of k-step pattern predictability.

Remark 1: Given system $\hat{G}$ and pattern Θ, let $\mu = 0$ and $k = 0$. In this case, k-step pattern predictability degenerates into pattern predictability. That is to say, the language L

generated by $\hat{G}$ being k-step pattern predictable w.r.t. Θ and μ is equivalent to the fact that the language L is pattern predictable w.r.t. Θ. Therefore, k-step pattern predictability generalizes the pattern predictability introduced by T. Jéron et al. in (Jéron, 2008).

The following example further explains the notion of k-step pattern predictability.

Example 2: Let us consider the plant $\hat{G}_2$ described in Fig.2, in which the set of events is $\Sigma = \{a,b,c,d,e\}$ and 0 is the initial state. Suppose that $k=1$ and $\mu=1$. Let $\Sigma_{uo} = \{c,e\}$ be the set of unobservable events and the pattern under consideration be the set $\Theta = \{cb, de\}$.

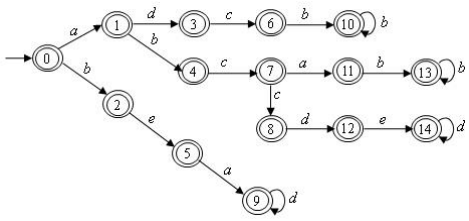

Fig. 2. Plant $\hat{G}_2$.

By a simple inspection of $\hat{G}_2$ in Fig.2, it is known that there are three branches which contain pattern string cb or de, i.e., $\wp = \{adcb^*, abcab^*, abccde^*\}$ and $\Psi_\wp(\Theta)$={adcb,*abcab*, *abccde*}.

Let's now show that for each trace s in $\Psi_\wp(\Theta)$, there exists a strict prefix uv and the length of v is greater than or equal to μ+k, such that each string u' in L that records the same observed behavior as u and its long enough continuation t, i.e. $u't$, is in $\wp$.

Let $s = adcb$(*abcab* or abccde) and $uv = adc$(*abca* or *abccd*). Clearly, $uv \in \bar{s}$. Pick $u = a$, $v = dc$(*bca* or *bccd*), then $\|v\| \geq \mu + k$ (i.e. $\|v\| \geq 2$). It can be inferred that $\chi_\mu^{-1}(\chi_\mu(u)) \cap L = \{a, ad, ab\}$, $L/a = \{dcb^*, bcab^*, bccded^*\}$, $L/ad = \{cb^*\}$ and $L/ab = \{cab^*, ccded^*\}$. For any $\mu' \in \{a, ad, ab\}$ and $t \in L/\mu'$, $\|t\| \geq n \Rightarrow (\mu' t \in \wp)$.

Based on the above analysis, it can be known that for any string $s \in \Psi_\wp(\Theta)$, there is a $u = a$ and $uv \in \bar{s}$ such that $\|v\| \geq \mu + k$ and $\mathbb{C}(u) = 1$. Therefore, the language L generated by $\hat{G}_2$ is one step pattern predictable w.r.t. Θ and μ. That is, after the observation of the event a, it is certain that the pattern cb or de will be occurred at least one step later in case of observation delay $\mu = 1$.

4. VERIFICATION OF K-STEP PATTERN PREDICTABILITY

In this section, an algorithm is developed to verify the property of k-step pattern predictability. In the work of (Jéron, 2008), an algorithm for verification of pattern predictability is based on the construction of a verifier automaton. However, the algorithm cannot be directly used to verify the property of k-step pattern predictability. Hence, this paper presents a novel polynomial-time verification of k-step pattern predictability based on the similar verifier (Jiang, 2001; Yoo, 2002). Furthermore, the new algorithm can be used to verify the pattern predictability proposed by T. Jéron et al.; see Example 5 for details.

4.1 Verification algorithm

Since the verification algorithm provided later requires some concepts, let's introduce them first.

Definition 7: Given two sets A and B, use $A-B$ to denote that the elements in set B are removed from the set A.

Definition 8: Le $G_A = (X_A, \Sigma_A, \delta_A, X_{\mathrm{m}}^A, X_0^A)$ and $G_B = (X_B, \Sigma_B, \delta_B, X_m^B, X_0^B)$ be two FSA. Define the product FSA of G_A and G_B as $G_A \times G_B = (X_{AB}, \Sigma_{AB}, \delta_{AB}, X_{\mathrm{m}}^{AB}, x_0^{AB})$, where $X_{AB} \subseteq X_A \times X_B$, $x_0^{AB} = (x_0^A, x_0^B)$ and $X_m^{AB} = \{(x_A, x_B) \in X_{AB} : x_A \in X_m^A \wedge x_B \in X_m^B\}$. For $\forall (x_A, x_B) \in X_{AB}$, the transition function $\delta_{AB}((x_A, x_B), \sigma) = (\delta_A(x_A, \sigma), \delta_B(x_B, \sigma))$ if $\delta_A(x_A, \sigma) \in X_A$ and $\delta_B(x_B, \sigma) \in X_B$; otherwise, undefined.

Definition 9: The state reached by a pattern trace is called a failure state; otherwise, is called a normal state. Denote the set of failure states by $\vartheta (\vartheta \subseteq X)$ and

$$\vartheta = \{x \in X : \exists s \in \psi_\wp(\Theta) \text{ s.t. } \delta(x_0, s) = x\}; \tag{5}$$

denote the set of normal states by X_N and $X_N = X - \vartheta$.

Definition 10: Given set ϑ, a subset of X_N, denoted by $\nabla(\vartheta)$, is defined as the set of states whose immediate successors belong to ϑ, i.e.,

$$\nabla(\vartheta) = \{x \in X_N : \exists \sigma \in \Sigma \text{ s.t. } \delta(x, \sigma) \in \vartheta\}. \tag{6}$$

Definition 11: For each state $x \in X_N$, $\lambda_{\min}(x)$ denotes the minimum number of steps required such that a pattern can occur from x, i.e.,

$$\lambda_{\min}(x) = \min_{s \in L(G,x) \wedge \delta(x,s) \in \nabla(\vartheta)} (s). \tag{7}$$

Note that $\lambda_{\min}(x)$ is undefined when patterns never happen from state x.

Definition 12: Given the value of μ and k, the boundary states, denoted by $\Omega_{\mu+k}$, is defined as the set of states in X_N from which after μ+k steps, a pattern will occur, i.e.,

$$\Omega_{\mu+k} = \{x \in X_N : \lambda_{\min}(x) = (\mu + k)\}. \tag{8}$$

Definition 13: Given set ϑ, let us define another subset of X_N, denoted by Λ. States belonging to Λ are approaching stable states from which there exists at least a trace cannot reach any failure state. This set is given by:

$$\Lambda = \{x \in X_N : \exists s \in \Sigma^* \text{ s.t. } (\forall n \geq 0) \wedge (\|s\| \geq n) \Rightarrow \delta(x,s) \notin \vartheta\}. \quad (9)$$

Now an algorithm is presented for verification of k-step pattern predictability.

Algorithm 1:

Given a networked DES $\hat{G} = (X, \Sigma, \delta, x_m, x_0, \mu)$ and a set of pattern strings $\Theta = \{s_1, s_2, ..., s_m\}$, where m is an integer.

Step 1: For each $i \in \{1,2,...,m\}$, construct automaton H_i for pattern string $s_i \in \Theta$.

Let $s_i = a_1 a_2 ... a_n \in \Sigma^*$ for an integer n, build a FSA $H_i = (X_{H_i}, \Sigma, \delta_{H_i}, x_m^{H_i}, x_0^{H_i})$, where $X_{H_i} = \{0,1,2,...,\|s_i\|\}$, $X_0^{H_i} = 0$, $x_m^{H_i} = \|s_i\|$ and for all $x \in (X_{H_i} - \{\|s_i\|\})$ and $\sigma \in \Sigma$,

$$\delta_{H_i}(x,\sigma) = \begin{cases} x+1 & \sigma = a_{x+1} \\ x & otherwise \end{cases}. \quad (10)$$

Step 2: Construct automaton $\hat{G}_H$ for recognizing the pattern traces of $\hat{G}$.

The automaton $\hat{G}_H$ is constructed by the synchronous product of $\hat{G}$ and H_i for $i = 1,2,...,m$, i.e.,

$$\hat{G}_H = \hat{G} \times H_1 \times ... \times H_m. \quad (11)$$

In $\hat{G}_H$, the strings that reach the marked states are the pattern traces of $\hat{G}$. Put them into set $\Psi_{\wp}(\Theta)$. Based on $\Psi_{\wp}(\Theta)$, compute the value of $l_{\min}(s)$.

Let's give a lemma first. The lemma reveals the relation of k-step pattern predictability and $l_{\min}(s)$.

Proposition 2: L generated by $\hat{G}$ is not k-step pattern predictable w.r.t. Θ and μ if $l_{\min}(s) \leq (\mu + k)$.

Proof: if $l_{\min}(s) \leq (\mu + k)$ means that there exists a $s \in \Psi_{\wp}(\Theta)$, but there is no $uv \in \overline{s}(uv \neq s)$ and $\|v\| \geq (\mu + k)$ such that $\mathbb{C}(u) = 1$. Therefore, L generated by $\hat{G}$ is not k-step pattern predictable w.r.t. Θ and μ.

Consider Example 2 again. It is already known that $\Psi_{\wp}(\Theta) = \{adcb, abcab, abccde\}$. Clearly, $l_{\min}(s)$=4. Suppose $\mu = 2$ and $k = 2$. $l_{\min}(s) \leq \mu + k$, thus L of $\hat{G}_2$ is not k-step pattern predictable w.r.t. Θ and μ.

Step 3: Establish the set of boundary states $\Omega_{\mu+k}$ and the set of approaching-stable states Λ.

According to the pattern traces of $\hat{G}$, compute ϑ and $\nabla(\vartheta)$. Then establish boundary states set $\Omega_{\mu+k}$ and approaching-stable states set Λ, respectively.

Step 4: Construct verifier automaton $V_{\hat{G}}$.

Based on the automaton $\hat{G}$, construct verifier automaton $V_{\hat{G}}$. The verifier automaton is a finite-state automaton

$$V_{\hat{G}} = (X_V, \Sigma, \delta_V, x_{V,0}). \quad (12)$$

where $X_V \subseteq X \times X$ is the set of states; $x_{V,0} = (x_0, x_0)$ is the initial state of the verifier; $\delta_V(x_v, \sigma)$ is the state transition function defined as follows:

If $\sigma \in \Sigma_{uo}$, then

$$\delta_V((x1,x2),\sigma) = \{(\delta(x_1,\sigma), x_2), (x_1, \delta(x_2,\sigma)), (\delta(x_1,\sigma), \delta(x_2,\sigma))\}.$$

If $\sigma \in \Sigma_o$, then

$$\delta_V((x1,x2),\sigma) = \{(\delta(x_1,\sigma)), \delta(x_2,\sigma)\}.$$

Step 5: Check if there exists a state in the $V_{\hat{G}}$ satisfying the condition (13) given by Theorem 1 below. If the answer is yes, then L is not k-step pattern predictable w.r.t. Θ and μ; otherwise, L is k-step pattern predictable w.r.t. Θ and μ.

Theorem 1: Let $l_{\min}(s) > (\mu + k)$ and $V_{\hat{G}}$ be the verifier of $\hat{G}$. L generated by $\hat{G}$ is not k-step pattern predictable w.r.t. Θ and μ iff there exists a state $(x_1, x_2) \in X_V$ in automaton $V_{\hat{G}}$ such that

$$[x_1 \in \Lambda] \wedge [x_2 \in \Omega_{\mu+k}]. \quad (13)$$

Proof: ($\Rightarrow$) Suppose that L is not k-step pattern predictable w.r.t. Θ and μ. There exists a string $s \in \Psi_{\wp}(\Theta)$ such that for all $uv \in \overline{s}(uv \neq s, \|v\| \geq (\mu + k))$, $\mathbb{C}(u) \neq 1$. That is to say, for all uv, there exist $u' \in (\chi_{\mu}^{-1}\chi_{\mu}(u)) \cap L)$, $t \in L/u'(\|t\| \geq n)$ such that $u't \notin \wp$. Let $\delta(x_0, u) = x_1$ and $\delta(x_0, u') = x_2$. According to the definition of boundary states set and approaching-stable states set, there are $x_1 \in \Omega_{\mu+k}$ and $x_2 \in \Lambda$. Thus, condition (13) holds.

($\Leftarrow$) Suppose that there exists a state $x = (x_1, x_2) \in X_V$ such that $x_1 \in \Lambda$ and $x_2 \in \Omega_{\mu+k}$. Let $s_0 \in L(V_{\hat{G}})$ be a string of $V_{\hat{G}}$ such that $\delta_V(x_{V,0}, s_0) = x$. Let $\delta(x_0, s_0) = x_2$. By the definition of $V_{\hat{G}}$, $\exists s \in L$ such that $P(s) = P(s_0)$ and $\delta(x_0, s) = x_1$. According the definition of approaching-stable states give by (9), $\exists t \in \Sigma$ such that $st \notin \wp$. By Definition 7, L is not k-step pattern predictable w.r.t. Θ and μ.

Remark 2: Let us discuss the complexity of the above proposed verification algorithm. For the first step, since the number of states of H_i is $|X_{H_i}|$, and the number of feasible transitions from every state is $|\Sigma|$, the construction of H_i takes $O(|X_{H_i}||\Sigma|)$ time. The time complexity of step 2 and step 3 are $O(|X_{H_i}||X||\Sigma|)$ and $O(|X||\Sigma|)$, respectively. The number

of states of $V_{\hat{G}}$ is in the worst case equal to $|X|^2$. Moreover, the maximum number of transitions of states in the $V_{\hat{G}}$ is equal to $|\Sigma|$. The time complexity of step 4 and step 5 is $O(|X|^2|\Sigma|)$. So the overall computational complexity of the approach is polynomial.

4.2 Illustrative examples

To illustrate the algorithm, following examples are presented.

Example 3: Consider the plant $\hat{G}_2$ shown in Fig.2 again, where c and e are unobservable events and pattern is $\Theta = \{cb, de\}$. Suppose $k = 1$ and $\mu = 1$. Example 2 has shown that L generated by $\hat{G}_2$ is k-step pattern predictable w.r.t. Θ and μ. Now verify this conclusion through Algorithm 1.

Let $s_1 = cb$ and $s_2 = de$. The first step is to construct automata H_i for pattern strings $s_i, i = 1, 2$, which are shown in Fig.3 and Fig.4, respectively. Note that state 2 is the marked state.

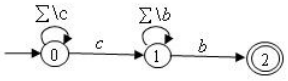

Fig. 3. Automaton H_1.

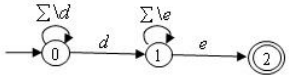

Fig. 4. Automaton H_2.

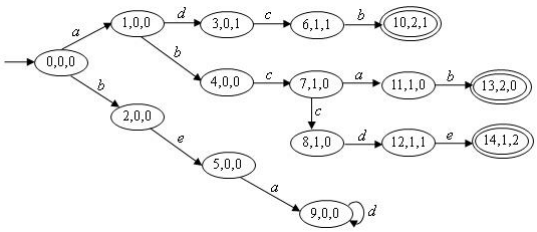

Fig. 5. Automaton $\hat{G}_{2H}$.

In the Fig.5, (10,2,1), (13,2,0) and (14,1,2) are marked states, which indicates that the strings that reach them are the pattern traces. Therefore, $\Psi_{\wp}(\Theta) = \{adcb, abcab, abccde\}$. Clearly, for $s \in \Psi_{\wp}(\Theta)$, $l_{\min}(s) = 4$ and $l_{\min}(s) > (\mu + k)$. By the pattern traces of $\Psi_{\wp}(\Theta)$, it is known that the set of failure states $\vartheta = \{10,13,14\}$. Further, $\nabla(\vartheta) = \{6,11,12\}$, the set of boundary states $\Omega_{\mu+k} = \{1,4,7\}$ and the set of approaching-stable states $\Lambda = \{0,2,5,9\}$.

Now construct verifier automaton $V_{\hat{G}_2}$, which is shown in Fig.6.

By a detailed inspection of Fig.6, it is known that $V_{\hat{G}_2}$ does not contain any state satisfying the condition (13). Thus, according to Theorem 1, L of $\hat{G}_2$ is k-step pattern predictable w.r.t. Θ and μ.

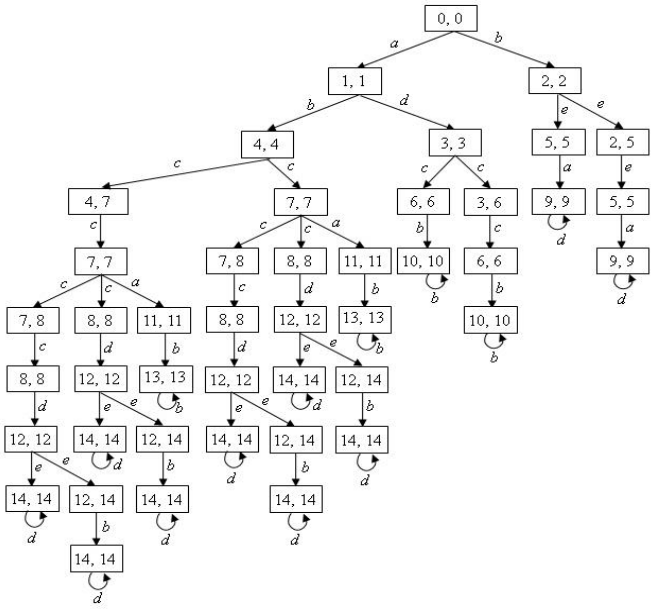

Fig. 6. Verifier automaton $V_{\hat{G}_2}$.

Example 3 considers the language L that is k-step pattern predictable w.r.t Θ and μ. The following example is a counter-example to show that L is not k-step pattern predictable w.r.t. Θ and μ.

Example 4: Consider the plant $\hat{G}_3$ shown in Fig.7, which is a variation of $\hat{G}_2$. Suppose that the pattern and $\Sigma_{uo} = \{c, e\}$. Let $k = 1$ and $\mu = 1$.

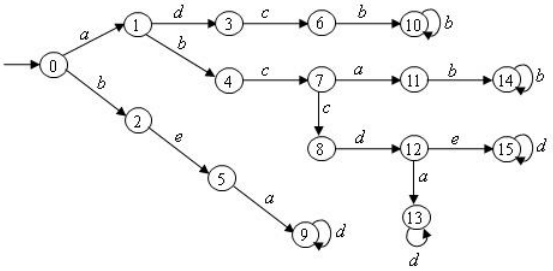

Fig. 7. $\hat{G}_3$ of Example 4.

In the following, the result that L of $\hat{G}_3$ is not k-step pattern predictable w.r.t. Θ and μ will be verified by Algorithm 1.

Automata H_1 and H_2 are also constructed as that in Fig.3 and Fig.4.The automaton $\hat{G}_{3H} = \hat{G}_3 \times H_1 \times H_2$ is described by Fig.8.

Note that in the $\hat{G}_{3H}$, marked states are (10,2,1), (14,2,0) and (15,1,2)}. By the marked states, it is known tha ϑ={10, 14,15} and $\nabla(\vartheta)$={6,11,12}. And $\Omega_{\mu+k} = \{1,4,7\}$, $\Lambda = \{0,1,2, 4,5,7,8,9,12,13\}$.

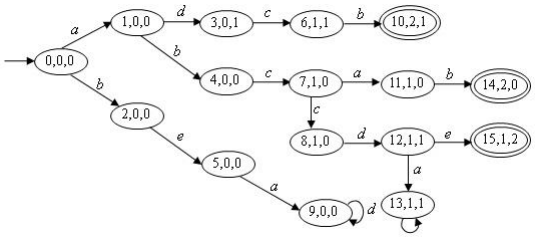

Fig. 8. Automaton $\hat{G}_{3H}$.

Verifier automaton $V_{\hat{G}_3}$ is constructed as that shown in Fig.9.

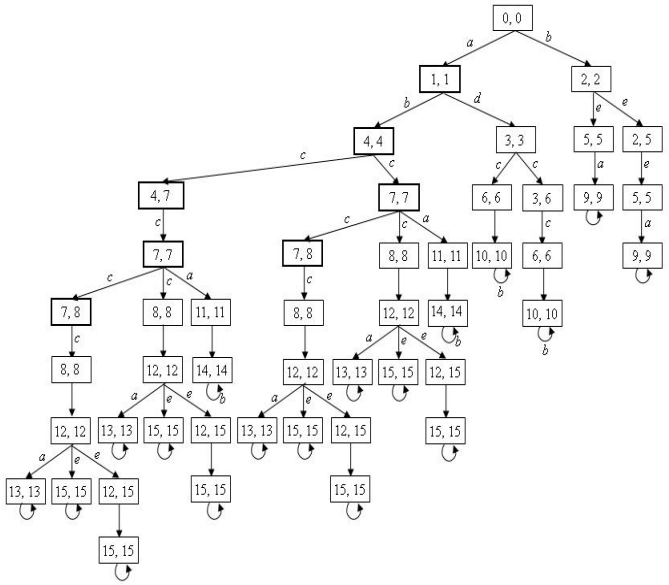

Fig. 9. Verifier Automaton $V_{\hat{G}_3}$.

In the $V_{\hat{G}_3}$, the states with thick borders are (1,1), (4,4), (4,7), (7,7) and (7,8), which satisfy the condition (13). Therefore, L of $\hat{G}_3$ is not k-step pattern predictable w.r.t. Θ and μ by Theorem 1.

Finally, another example is given to illustrate the algorithm can also be used to verify the pattern predictability proposed by T. Jéron et al.

Example 5: Consider a variation of Example 1 shown in Fig.10. $\Theta = \{f_1 f_2, f_2 f_1\}$ and unobservable events are the same as in Example 1.

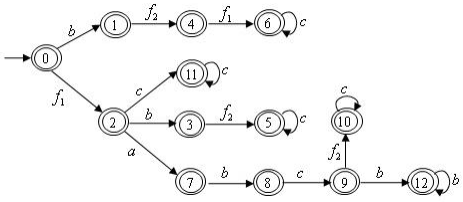

Fig. 10. $\hat{G}_4$ of Example 5.

It is known that the language L of $\hat{G}_4$ in Fig.10 is not pattern predictable w.r.t. Θ in (Jéron, 2008). In the following, the result will be verified again by Algorithm 1.

Since the system has no communication delays and does not consider advance prediction, let $\mu = 0$ and $k = 0$.

Automaton H_1 for pattern string $f_1 f_2$ and automaton H_2 for $f_2 f_1$ are constructed as that in Fig.11 and Fig.12. The automaton $\hat{G}_{4\mathrm{H}} = \hat{G}_4 \times H_i$, i = 1,2, is constructed as Fig.13 shows.

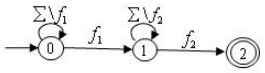

Fig. 11. H_1 of Example 5.

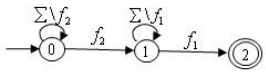

Fig. 12. H_2 of Example 5.

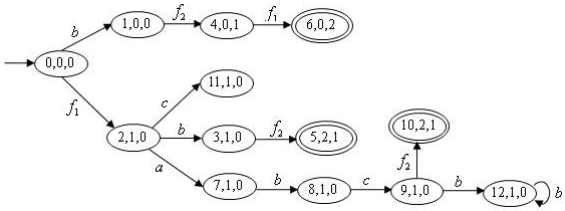

Fig. 13. $\hat{G}_{4H}$ of Example 5.

As Fig.13 shows, (6,0,2), (5,2,1) and (10,2,1) are marked states. Therefore, ϑ={6,5,10} and $\nabla(\vartheta)$={4,3,9}. And there are $\Omega_{\mu+k} = \{4,3,9\}$ and Λ={0,2,7,8,9,11,12}.

Verifier automaton $V_{\hat{G}_4}$ is shown in Fig.14.

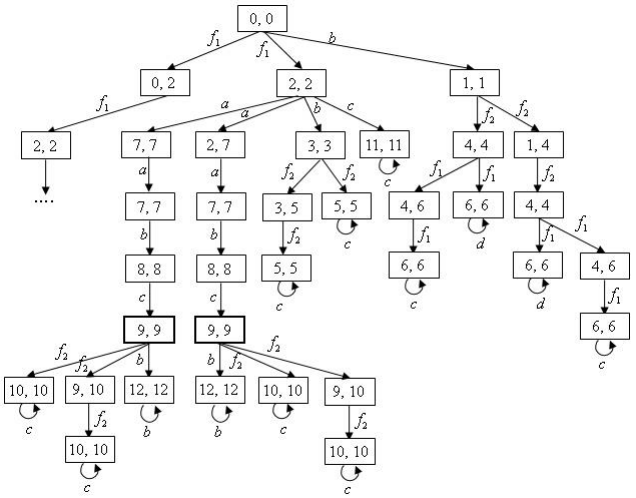

Fig. 14. Verifier automaton $V_{\hat{G}_4}$.

In the Fig.14, (9,9) is a state with the thick border. As is known that, 9 is not only a boundary state, but also an approaching-stable state, i.e., $(9 \in \Lambda) \wedge (9 \in \Omega_{\mu+k})$. This satisfies the condition (13). Therefore, L of $\hat{G}_4$ is not k-step pattern predictable w.r.t. Θ and μ by Theorem 1. That is, $\hat{G}_4$ is not pattern predictable.

5. THE PREDICTOR

In this section, this paper will explain how to construct a predictor that can be used to predict the occurrence of a pattern on line. In the work of (Jéron, 2008), a similar predictor was designed. However, the predictor cannot be directly used to predict the occurrence of a pattern in a networked DES with communication delays. Thus, it is necessary to design a new predictor for the networked DES so that it cannot only predict the occurrence of a pattern, but also predict exactly how many steps the system may perform before it occurs.

The construction of a predictor is completed in two steps.

The first step is to attach labels to the states of automaton $\hat{G}$. The details are as follows:

1) Attach N label to initial state and all approaching-stable states.

2) Attach F label to failure states.

3) Attach $F+$ label to the states of $\nabla(\vartheta)$.

4) For each $i \in \{0,1,...,(\mu+k-1)\}$, attach label $F+(\mu+k-i)$ to the states of $\Omega_{\mu+k-i}$.

Here, N means that no patterns occur, the system is in a normal state; F means that a pattern string has occurred and the system is in a failure state; $F+$ indicates that the system is in a normal state, but a pattern string will occur immediately; $F+(\mu+k-i)$ indicates that the system is in a normal state, but a pattern string will occur after $(\mu+k-i)$ steps.

The second step is to construct the standard observer automaton as follows:

$$Obs(\hat{G}) = (X_{obs}, \Sigma_o, \delta_{obs}, x_0^{obs}). \tag{12}$$

where $X_{obs} \subseteq 2^X$ is the set of states, Σ_o is the set of events and $\delta_{obs} : X_{obs} \times \Sigma_o \to X_{obs}$ is the transition function. It is defined by: for any $x \in X_{obs}$ and $\sigma \in \Sigma_o$,

$$\delta_{obs}(x,\sigma) = \Re(x,\sigma). \tag{13}$$

$x_0^{obs} \in X_{obs}$ is the initial state defined by $x_0^{obs} = \Re(x_0, \varepsilon)$.

Remark 3: Let us discuss the complexity of the above proposed construction procedure. For the first step, since the number of states of $\hat{G}$ is $|X|$, and the number of feasible transitions from every state is $|\Sigma|$, attaching labels to $\hat{G}$ takes $O(|X||\Sigma|)$ time. The numbers of states of $Obs(\hat{G})$ is in the worst case equal to $|2^X|$. Moreover, the maximum number of transitions of states in the $Obs(\hat{G})$ is equal to $|\Sigma|$. The time complexity of step 2 is $|2^X| \times |\Sigma|$. So the overall computational complexity of the approach is exponential.

Example 6: Once again, $\hat{G}_2$ of Example 2 is used to illustrate the procedure described above.

From example 3, the initial state is 0 and approaching-stable states are 0, 2, 5 and 9, attaching N label to them; Failure states are 10, 13 and 14, attaching F label to them; 6, 11 and 12 are the states of $\nabla(\vartheta)$, attaching $F+$ label to them; 1, 4 and 7 are the states of $\Omega_{\mu+k}$, attaching $F+(\mu+k)$ i.e. $F+2$ label to them; 3, 7 and 8 are the states of $\Omega_{\mu+k-1}$, attaching $F+(\mu+k-1)$ i.e. $F+1$ label to them.

The labeled $\hat{G}_2$ is shown in Fig.15.

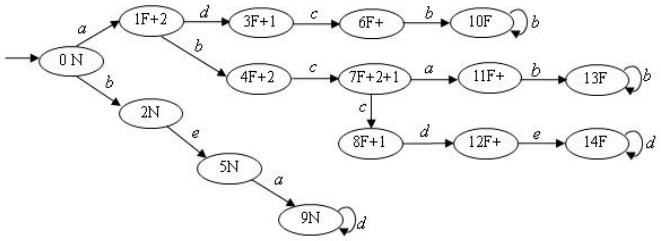

Fig. 15. The labeled $\hat{G}_2$.

Based on the labeled $\hat{G}_2$, a predictor is constructed as that in Fig.16.

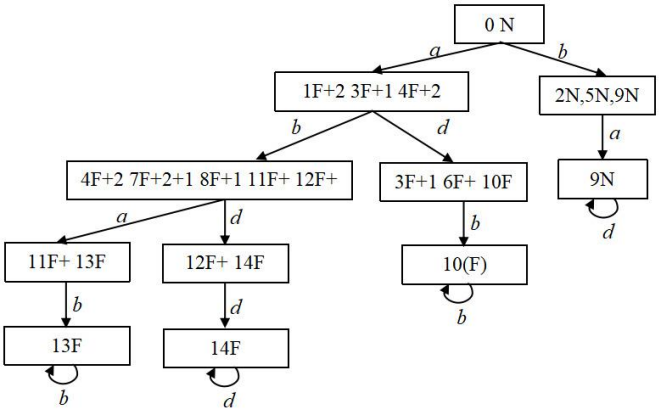

Fig. 16. The predictor of $\hat{G}_2$.

When the predictor runs on-line and event a is observed, the system may be in 1, 3 or 4 state. The implication of label $1F+2$ is that if the system is in state 1, then a pattern string will occur after two steps; label $3F+1$ means that if the system is in state 3, then a pattern string will occur after one step; label $4F+2$ means that if the system is in state 4, then a pattern string will occur after two steps. Based on the above analysis, it can be determined that a pattern string will occur at least after one step.

6. CONCLUSIONS

This paper presented new results on pattern predictability of networked DESs. The new notion of k-step pattern predictability is defined and an off-line algorithm is described to verify the property. It is shown that the off-line algorithm is polynomial in the number of states of the system.

Moreover, a predictor is designed to predict the occurrence of a pattern during the operation of the system. In future work, the pattern predictability of fuzzy networked DESs or stochastic networked DESs will continue to be considered.

ACKNOWLEDGEMENTS

This work was supported by "the National Natural Science Foundation of China" (grant number 61673122) and "the Natural Science Foundation of Guangdong Province" (grant number 2019A1515010548) of China.

REFERENCES

Benmessahel, B., Touahria, M. and Nouioua, F. (2017). Predictability of fuzzy discrete event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 27(4), 641-673.

Chang, M., Dong, W., Ji, Y. and Tong, L. (2013). On fault predictability in stochastic discrete event systems. *Asian Journal of Control*, 15(5), 1458-1467.

Genc, S. and Lafortune, S.(2006). Diagnosis of patterns in partially-observed discrete-event systems. *In 45th IEEE Conference on Decision and Control*, San Diego, CA, December, 422-427.

Genc, S. and Lafortune, S. (2009). Predictability of event occurrences in partially-observed discrete-event systems. *Automatica*, 45, 301-311.

Geng, X., Ouyang, D. and Jiang, Z. (2020). Pattern diagnosis for stochastic discrete event systems. *Engineering Application of Artificial Intelligence*, 87, 1-10.

Jéron, T. , Marchand, H., Pinchinat, S., Cordier, M. (2006). Supervision patterns in discrete event systems diagnosis. in *Internat Workshop on Discrete Event System*, 262-268.

Jéron, T., Marchand, H., Genc, S. and Lafortune, S. (2008). Predictability of sequence patterns in discrete event systems. in Proc. *17th IFAC world conference*, July 6-11, 537-543.

Jiang, S., Huang, Z., Chandra, V., and Kumar, R. (2001). "A polynomial time algorithm for diagnosability of discrete event systems*", IEEE Trans. on Automatic Control*, vol. 46, no.8, pp.1318-1321.

Lamperti, G. and Zhao, X. (2017). Diagnosis of active systems by semantic patterns. *IEEE Trans. Syst. Man Cybern. Syst.*, 44(8), 1028-1043.

Lin, F. (2014). Control of networked discrete event systems: Dealing with communication delays and losses. *SIAM J.Control Optimiz*, 52(2), 1276-1298.

Liu, F. and Lin, H. (2009). Reliable decentralized supervisory control of discrete event systems with communication delays.*2009 IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics*, 368-373.

Liu, F. (2019). Predictability of failure event occurrences in decentralized discrete-event systems and polynomial time verification. *IEEE Trans. on Automation science and engineering*, 16(1), 498-504.

Park, S.-J. and Cho, K.-H. (2007). Supervisory control of discrete event systems with communication delays and partial observations. *Syst. Control Lett.*, 106-112.

Shu, S. and Lin, F. (2015). Supervisor synthesis for networked discrete event systems with communication delays. *IEEE Trans. on Automatic Control*, 60(8), 2183-2188.

Ye, L., Dague, P. and Yan, Y. (2009). An incremental approach for pattern diagnosability in distributed discrete event systems. In: *Proceedings 21st IEEE International Conference on Tools with Artificial Intelligence*, 123-130.

Ye, L. and Dague, P. (2012). A general algorithm for pattern diagnosability of distributed discrete event systems. *IEEE International Conference on Tools with Artificial Intelligence*, 130-137.

Yin, X. and Li, Z. (2016). Reliable decentralized fault prognosis of discrete-event systems. *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, 46(11), 1598-1603.

Yin, X. and Li, Z. (2019). Decentralized fault prognosis of discrete-event systems using state-estimate-based protocols. *IEEE Transactions on Cybernetics*, 49(4), 1302-1313.

Yoo, T. and Lafortune, S. (2002). Polynomial-time verification of diagnosability of partially-observed discrete-event systems. *IEEE Trans. on Automatic Control*, vol. 47(9), 1491-1495.

Zgorzelski, M. and Lunze, J. (2018). A new approach to tracking control of networked discrete-event systems. *International Federation of Automatic Control Hosting by Elsevier Ltd.*, 448-455.

Zhao, R., Liu, F. and Tan, J. (2019). Relative predictability of failure event occurrences and its opacity-based test algorithm. *International Journal of Control*, 92(7), 1600-1608.