Research Article

Fault Diagnosis for Discrete Event Systems Using Partially Observed Petri Nets

Ye Dandan ¹, Luo Jiliang,² and Su Hongye¹

¹State Key Laboratory of Industrial Control Technology and Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China

²College of Information Science and Engineering, Huaqiao University, Xiamen, China

Correspondence should be addressed to Ye Dandan; 11532024@zju.edu.cn

Received 23 March 2020; Accepted 14 May 2020; Published 26 June 2020

Academic Editor: Sajad Azizi

Copyright © 2020 Ye Dandan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This study proposes a fault diagnosis method of discrete event systems on the basis of a Petri net model with partially observable transitions. Assume that the structure of the Petri net model and the initial marking are known, and the faults can be modeled by its unobservable transitions. One of the contributions of this work is the use of the structure information of Petri net to construct an online fault diagnoser which can describe the system behavior of normal or potential faults. By modeling the flow of tokens in particular places that contain fault information, the variation of tokens in these places may be calculated. The outputs and inputs of these places are determined to be enabled or not through analyzing some special structures. With the structure information, traversing all the states is not required. Furthermore, the computational complexity of the polynomial allows the model to meet real-time requirements. Another contribution of this work is to simplify the subnet model ahead of conducting the diagnostic process with the use of reduction rules. By removing some nodes that do not contain the necessary diagnostic information, the memory cost can be reduced.

1. Introduction

Fault diagnosis is a critical issue in most industrial systems when preserving the safety of equipment and human operators. This issue has been studied in numerous studies that concern discrete event systems [1–6]. Once a fault has been detected and identified, the control law needs to be modified to safely advance the operations.

A discrete event system (DES) is a dynamic system which is driven by events. When a signal of a sensor or a movement of an executive device in the system is detected, the state of DES can be transformed automatically. Nowadays, DESs exist in so many intelligent systems including robot system, manufacturing system, and transportation system.

In some previous literature, the fault diagnosis of DESs is discussed by several approaches based on models [7–10], such as automata models, which usually lead to constructing a diagnoser automaton. Moreover, the diagnoser can be applied to analyze some diagnosable properties of systems,

i.e., to check whether detecting the occurrences of unobservable events associated with faults is possible by observing words with finite lengths.

Although automata models are suitable for describing DESs, the size of the system would limit their application. The models explicitly determine all possible states and thus would result in some quite large models when the size of the systems grows. To avoid enumerating system states and the consequent enlargement of states, Petri nets are exploited to address fault diagnosis given their merits of graphical structure. Most of the recent studies on fault diagnosis are based on the analysis of Petri net reachability graph [11–14], the direct properties of Petri nets [15, 16], and structural analysis of Petri nets [17, 18].

Given that the approach proposed in the present work is based on Petri nets, the past related studies are briefly recalled. Giua and Seatzu [19] presented a diagnostic approach to avoid the exhaustive enumeration of DES states and introduced basis marking and justifications, in which

the markings consistent with actual observations are characterized and the set of unobservable transitions that enables the characterizations are established. Cabasino et al. [20] developed similar approaches, such as the modified basis reachability graph and the basis reachability diagnoser, to compress the construction of a state space of bounded Petri nets. Basile et al. [21] forwarded an approach for checking K-diagnosability with the use of the technique of the integer linear programming. Liu et al. [22] proposed a technique called an on-the-fly incremental diagnostic. It can be applied to analyze both the diagnosability and K-diagnosability of bounded and also live Petri nets. Ben et al. [23] used T-invariants to define the priorities of branch investigation and rapidly determine the existence of an indeterminate cycle. Hadjicostis and Verghese [24] introduced system redundancy to detect and then isolate the fault markings. Prock [25] proposed an online technique for fault diagnosis. It can monitor the number of tokens within the P-invariants. If the total number of tokens within P-invariants is changed, then the error can be detected. Ru et al. [26] presented a method to address DESs by conducting the partial observation of states and events on the basis of the transformation of partially observed Petri nets into labeled Petri nets. Dotoli et al. [12] presented an algorithm based on the definition and solution of integer linear programming problems. The algorithm was used to characterize the fault-behavior properties needed by the system to reduce online computational effort. Fabre et al. [27] proposed a net-unfolding approach to design an online asynchronous diagnoser that can avoid state explosion. However, the online computational effort of the proposed approach is high because of the online construction of the Petri net structure by means of the unfolding.

It is easy to figure out that most of the previous methods cannot avoid the state explosion problem. The computational complexities of the proposed algorithms are exponential, and they are not suitable for online use with large intelligent systems.

For avoiding the state explosion problem, an online fault diagnosis strategy is proposed in this paper on the basis of the structure information of partially observed Petri nets. A number of reduction rules are adopted to simplify the construction of special structures without having to change the diagnosability property of the system. More precisely, assuming that the structure of the Petri net model and the initial marking are known, the faults can be modeled by their unobservable transitions. Furthermore, any additional unobservable transition may be associated with system legal behavior. Then, an algorithm decides whether the system behavior is normal or has potential faults when observed sequences occur, and special paths are defined to depict the structure information of Petri nets. Furthermore, subnets that contain fault information are constructed to describe the flow situation of the tokens. Based on these tokens, some of the inputs or outputs of particular places, which are the inputs or outputs of fault transitions, are considered for enabling. Finally, the fault diagnoser determines the inherent fault behavior.

The main advantages of the present work are as follows. Compared with the work of Fabre et al. [27], our algorithm

does not require offline calculations using the structure information of Petri nets. Our method is also more likely applicable than that of Fabre et al. [27] for minimizing memory cost with the reduction rules. The diagnostic algorithm in this paper is with great advantage for avoiding the state explosion especially when looking for a reasonably efficient method for online use with large intelligent systems.

The remainder of the paper is structured as follows. Section 2 provides the basic definitions and notations. Section 3 presents the special structures used to describe the inherent fault information and the reduction rules applied to simplify the model. Section 3.5 specifies the algorithm for online fault diagnosis and proposes the fault diagnoser. Section 4 gives an example of an intelligent warehouse center to verify the algorithm. Section 5 draws the conclusions.

2. Preliminaries

This section introduces the basic characteristics of Petri nets. For the detailed discussion on Petri nets, refer to [28].

2.1. Basic Petri Net Notations

Definition 1. A Petri net is a structure that can be described as $N = (P, T, \mathbf{Pre}, \mathbf{Post})$, where

- (i) P is the finite set of places
- (ii) T is the finite set of transitions, and $P \cap T = \emptyset$
- (iii) $\mathbf{Pre}: P \times T \longrightarrow \mathbb{N}$ is the *pre*-incidence function, where $\mathbb{N}$ is the set of natural numbers
- (iv) $\mathbf{Post}: T \times P \longrightarrow \mathbb{N}$ is the *post*-incidence function, where $\mathbb{N}$ is the set of natural numbers

The symbol $\bullet p(\bullet t)$ is used for the *pre*-set of place $p \in P$ (transition $t \in T$) and $p^\bullet(\bullet t)$ is used for the *post*-set of place $p \in P$ (transition $t \in T$), respectively; e.g., $t^\bullet = \{p \in P \mid \mathbf{Post}(p, t) \neq 0\}$.

The marking M of Petri net N is represented by $|P|$ -vector, in which in each place p of P , a nonnegative number of tokens $M: P \longrightarrow \mathbb{N}$ are assigned. Then, $M(p)$ is the number of tokens present in p with the marking M .

A Petri net system is a pair $\langle N, M_0 \rangle$ where N is a connected graph with at least one place and one transition, while M_0 is an initial marking of N . $R(N, M_0)$ denotes the set of reachable markings of the net system $\langle N, M_0 \rangle$.

Given the Petri net N and marking M , if and only if $\forall p \in \bullet t$ and $M(p) \geq \mathbf{Pre}(p, t)$, then we can say the transition $t \in T$ is enabled in M . It is denoted as $M[t\rangle$. A transition t that is enabled may be fired to derive the marking $M' = M + \mathbf{C}(\cdot, t)$, where $\mathbf{C} = \mathbf{Post} - \mathbf{Pre}$, and it is denoted as $M[t\rangle M'$.

A firing sequence from M is a sequence of transitions $\sigma = t_1 t_2 \cdots t_k$, such that $M[t_1\rangle M_1[t_2\rangle M_2 \cdots [t_k\rangle M_k$, which is denoted as $M[\sigma\rangle M_k$. T^* is a set of all sequences in a Petri net. An enabled sequence σ is denoted as $M[\sigma\rangle$, while $t_i \in \sigma$ implies that transition t_i belongs to sequence σ . If and only if there exists a sequence σ and it satisfies $M_0[\sigma\rangle M'$, then we can say the marking M' is reachable from the initial marking M_0 .

Set T can be partitioned into disjointed sets of observable transitions (represented by filled sticks) and unobservable transitions (represented by empty sticks) referred to as T_o and T_{uo} , respectively. An observed sequence is $\omega = t_1 t_2 \dots t_k$, where $t_1, t_2, \dots, t_k \in T_o$. In this paper, the fault events $t_f \in T_f$ are supposed to be unobservable, i.e., $T_f \subseteq T_{uo}$.

Definition 2 (see [26]). A partially observed Petri net G is a 3-tuple (N, P_o, T_o) , where

- (i) N is a Petri net with n places and m transitions
- (ii) P_o is the set of observable places with $P_o \subset P$
- (iii) T_o is the set of observable transitions with $T_o \subset T$

In this paper, we assume the set $P_o = \emptyset$, which means all the places are unobserved.

2.2. Subnets and Projections. In this part we present the definitions of subnets and projections.

Definition 3 (see [29]). Given a Petri net $N = (P, T, \mathbf{Pre}, \mathbf{Post})$, $N' = (P', T', \mathbf{Pre}', \mathbf{Post}')$ is a T' -induced subnet on N , where $T' \subseteq T$, $P' = \bullet T' \cup T'^\bullet$, and $\mathbf{Pre}'$ and $\mathbf{Post}'$ are the restrictions of $\mathbf{Pre}$ and $\mathbf{Post}$ to P' and T' , respectively.

Definition 4 (see [30]). Given a Petri net N , a path is an oriented sequence which is alternately comprised of the nodes of the Petri net N , denoted as π .

Consider a subnet N' and a path π in a Petri net. π is outside of N' if and only if all nodes of π do not belong to N' , denoted as $\pi \neq \ll N'$.

Definition 5. Given a set of transitions R and a sequence σ in a Petri net, $\rho(\sigma, R)$ is the projection of transitions in σ on R .

Example 1. If $\sigma = t_1 t_2 \varepsilon_3 t_4 \varepsilon_5 t_6$, where $t_1 t_2 t_4 t_6$ are observable and $\varepsilon_3 \varepsilon_5$ are unobservable, and set $R = T_o$, then $\rho(\sigma, R) = t_1 t_2 t_4 t_6$.

2.3. Basic Assumptions. In this part, some assumptions exploited in this study are presented in advance.

Assumption 1. No cycle of unobservable transitions exists.

Assumption 2. The Petri net N that models the DES and the initial marking $\mathbf{m}_0$ are known.

Assumption 3. Once a fault occurs, the system would remain to be faulty infinitely. Then, we call these faults permanent.

Assumption 1 is commonly adopted in the field of fault diagnosis of Petri net models, whereas Assumptions 2 and 3 correspond to levels of system knowledge.

3. Special Structures and Reduction Rules

In this section, we focus on some special structures of Petri net and its reduction rules.

3.1. Special Structures

Definition 6. Given a place p and a path in a Petri net model, this path is defined as an observable path of this place p , denoted as π^p , if it satisfies the following:

- (i) Its beginning node (or end node) is place p and its end node (or beginning node) is an observable transition
- (ii) The rest of its transitions are unobservable

Definition 7. Given a place p in a Petri net N , if π^p exists, an observed subnet of place p is denoted as SN^p which is only comprised of all π^p .

Consider a partially observed Petri net N and a place p . Assume that the number of nodes in N is $|m + n|$ and the number of nodes in SN^p is $|r|$. Obviously $|r| \leq |m + n|$ and the computational complexity of constructing SN^p is polynomial.

Definition 8. Given a place p in a Petri net N , a SN^p is defined as a diagnosable subnet of place p , denoted as DSN^p , if it satisfies the following:

- (i) Any places inside SN^p cannot be connected to any transitions outside SN^p
- (ii) Any unobservable transitions inside SN^p cannot be the inputs of any places outside SN^p

Example 2. The yellow path $p_4 \varepsilon_3 p_3 t_2$ is an observable path of place p_1 , namely, π^{p_1} , as shown in Figure 1.

Example 3. Figure 2 is an observed subnet of place p_4 , namely, SN^{p_4} . It is also a DSN^{p_4} .

For a fault transition t_f , its inputs and outputs are $\bullet t_f$ and $t_f^\bullet$. If a place $p \in \bullet t_f$ (or $p \in t_f^\bullet$), then we use $\vec{p}_f$ (or $\bar{p}_f$) for brief writing.

Definition 9. Given a fault transition t_f , a path $\pi^{\bar{p}_f}$ is called a fault observable path of $\bar{p}_f$, if it satisfies the following:

- (i) The beginning node of $\pi^{\bar{p}_f}$ is $\bar{p}_f$
- (ii) $\pi^{\bar{p}_f}$ contains t_f

Definition 10. Given a fault transition t_f , a path $\pi^{\vec{p}_f}$ is called a fault observable path of $\vec{p}_f$, if it satisfies the following:

- (i) The beginning node of $\pi^{\vec{p}_f}$ is $\vec{p}_f$
- (ii) $\pi^{\vec{p}_f}$ contains t_f

Definition 11. Given a $\text{DSN}^{\bar{p}_f}$ in a Petri net N , $\text{FDSN}^{\bar{p}_f}$ is called a fault-diagnosable subnet of place $\bar{p}_f$, if it satisfies the following:

- (i) $\text{FDSN}^{\bar{p}_f}$ is a part of $\text{DSN}^{\bar{p}_f}$
- (ii) $\pi^{\bar{p}_f}$ does not belong to $\text{FDSN}^{\bar{p}_f}$

FIGURE 1: A partially observed Petri net.

FIGURE 2: An observed subnet of place p_4 .

Definition 12. Given a $DSN^{\bar{p}_f}$ in a Petri net N , $FDSN^{\bar{p}_f}$ is called a fault-diagnosable subnet of place $\bar{p}_f$, if it satisfies the following:

- (i) $FDSN^{\bar{p}_f}$ is a part of $DSN^{\bar{p}_f}$
- (ii) $\pi^{\bar{p}_f}$ does not belong to $FDSN^{\bar{p}_f}$

Example 4. Place p_4 is the output of fault f_1 and Figure 3 is a fault-diagnosable subnet $FDSN^{p_4}$.

3.2. Reduction Rules. The basic structures of Petri nets are and-joint, and-split, or-joint, or-split, loop, and sequence, as shown in Figure 4. Some reduction rules based on these structures were already proposed in [31, 32]. However, in the rule (i) in [31, 32], the authors did not consider the inherent fault information on the unobservable transition, and the rules (ii) and (iii) are not mentioned.

However, not all rules can preserve the diagnosability property. To overcome this limitation, a few other simple rules are proposed as shown in Figure 5.

3.2.1. Fusion of Places in Vertical (Figure 5(a)). If there exists a transition between two places that is normal and unobservable, and it has one input and output, then this transition can be omitted and two places can be merged into a new place for reducing the number of nodes in the net. The marking of the new place is the sum of the markings of two previous places.

FIGURE 3: A fault-diagnosable subnet of p_4 .

FIGURE 4: Six structures of Petri net.

3.2.2. Fusion of Transitions in Parallel (Figure 5(b)). If there are several transitions with the same one input and output, and they are all normal and unobservable, then these transitions can be merged into one transition that is normal and unobservable. The new initial marking stays the same.

3.2.3. Fusion of Places in Parallel (Figure 5(c)). If there are several places with the same one input and output, then these places can be merged into one place. The marking of this place is the minimal of the several ones.

Example 5. As shown in Figure 3, $FDSN^{p_4}$ and f_2 are a fault transition that contains fault information. By using reduction rule i), places p_7 and p_2 are merged and transition ϵ_3 can be suppressed. Thus, transition f_2 cannot be reduced because it contains fault information. The reduced model is shown in Figure 6.

Theorem 1. With the use of the above reduction rules, the diagnosability of the reduced Petri net is consistent with that of the initial version.

FIGURE 5: Reduction rules applied to Petri net.

FIGURE 6: The reduced Petri net model.

Proof. If the initial Petri net is not diagnosable, then the two sequences of σ_1 and σ_2 exist with the same observation. Furthermore, faults exist in σ_2 but not in σ_1 . After the fault occurs, σ_2 can be arbitrarily long. Let a regular and also unobservable transition ε be contained in $\sigma_1(\sigma_2)$, and this transition ε can be removed with the use of some previous reduction rules. The observation of this reduced sequence $\sigma_1^*(\sigma_2^*)$ is retained even removing the transition ε . Therefore, the system is still not diagnosable.

If the initial Petri net is diagnosable, then no two sequences of σ_1 and σ_2 exist with the same observation. Furthermore, faults exist in σ_2 but not in σ_1 . After the fault occurs, σ_2 can be arbitrarily long. Let a regular and also unobservable transition ε be contained in $\sigma_1(\sigma_2)$, and this transition ε can be removed with the use of some previous reduction rules. The observation of this reduced sequence

$\sigma_1^*(\sigma_2^*)$ is retained (i.e., not observable). Therefore, the system is still diagnosable. $\square$

3.3. Level Functions

Definition 13. Given two transitions t_1 and t_2 in a Petri net system, if $\bullet t_1 = t_2^\bullet$, transition t_1 is called the up-transition of t_2 , and transition t_2 is called the down-transition of t_1 .

Definition 14. Given a path π^p , if its end node is place p , this path is called the up-observable path of p , denoted as π_u^p .

Definition 15. Given a path π^p , if its beginning node is p , this path is called the down-observable path of p , denoted as π_d^p .

With the reduction rules, several unobservable transitions without fault information may be reduced. However, some unobservable transitions cannot be reduced. In reduction rule (i), if transition ε contains fault information, then places p_i and p_j cannot be merged. Thus reduction rule (i) needs to be modified. If p_i is the output or input of a fault transition, then the new marking in the reduced model is $\hat{M}(p_i) = M(p_i)$ and $\hat{M}(p_j) = 0$. If p_j is the output or input of the fault transition, then the new

marking in the reduced model is $\hat{M}(p_i) = M(p_j)$ and $\hat{M}(p_j) = 0$.

Definition 16. Given a fault transition t_f and a place p , $\hat{T}(p)$ is called the up-transition set of place p if all the transitions of $t \in \hat{T}(p)$ satisfy $t \in \pi_u^p$.

Definition 17. Given a fault transition t_f and a place p , $\check{T}(p)$ is called the down-transition set of place p if all the transitions of $t \in \check{T}(p)$ satisfy $t \in \pi_d^p$.

Definition 18. Given a FDSN $\vec{P}_f$, $K: \hat{T}(\vec{p}_f) \rightarrow \mathbb{N}$ is called the up-level function of $\vec{p}_f$ and for all $t \in \hat{T}(\vec{p}_f)$ satisfies

$$K(t) = \begin{cases} 0, & \text{if } t \in T_o, \\ K(t') + 1, & \text{if } t \in T_u \text{ and } t' \in \hat{T}(\vec{p}_f). \end{cases} \quad (1)$$

And t' is the up-transition of t .

Definition 19. Given a FDSN $\vec{P}_f$, $K: \check{T}(\vec{p}_f) \rightarrow \mathbb{N}$ is called the up-level function of $\vec{p}_f$ and for all $t \in \check{T}(\vec{p}_f)$ satisfies

$$K(t) = \begin{cases} 0, & \text{if } t \in T_o, \\ K(t') + 1, & \text{if } t \in T_u \text{ and } t' \in \check{T}(\vec{p}_f). \end{cases} \quad (2)$$

And t' is the up-transition of t .

Definition 20. Given a FDSN $\vec{P}_f$, $K: \check{T}(\vec{p}_f) \rightarrow \mathbb{N}$ is called the down-level function of $\vec{p}_f$ and for all $t \in \check{T}(\vec{p}_f)$ satisfies

$$H(t) = \begin{cases} 0, & \text{if } t \in T_o, \\ H(t') + 1, & \text{if } t \in T_u \text{ and } t' \in \check{T}(\vec{p}_f). \end{cases} \quad (3)$$

And t' is the down-transition of t .

Definition 21. Given a FDSN $\vec{P}_f$, $K: \check{T}(\vec{p}_f) \rightarrow \mathbb{N}$ is called the down-level function of $\vec{p}_f$ and for all $t \in \check{T}(\vec{p}_f)$ satisfies

$$H(t) = \begin{cases} 0, & \text{if } t \in T_o, \\ H(t') + 1, & \text{if } t \in T_u \text{ and } t' \in \check{T}(\vec{p}_f). \end{cases} \quad (4)$$

And t' is the down-transition of t .

Based on Definitions 13 to 21, the transitions in FDSN $\vec{P}_f$ (or FDSN $\vec{P}_f$) can therefore be classified.

3.4. Maximal Number of Flow-In and Minimal Number of Flow-Out

Definition 22 (see [19]). Given an observed sequence ω in a Petri net, set $M_\omega = \{M \mid M_0[\sigma]M, \rho(\sigma, T_o) = \omega, \sigma \in T^*\}$ is called a consistent marking set of ω .

Definition 23. Given an observed sequence ω in a Petri net, if $\exists \omega' = \rho(\hat{T}(p), \omega)$, the maximal number of flow-in of place p with ω is $\theta(p, \omega) = \max[M(p) - M_0(p)]$, and $M \in M_{\omega'}$.

Consider a place p and an observed sequence ω , and assume the maximal capacity of p is k . The maximal number of flow-in of p must satisfy $\theta(p, \omega) \leq k$.

Algorithm 1 is used for calculating the maximal number of flow-in of place $\vec{p}_f$ (or $\vec{p}_f$).

Algorithm 1 is proposed on the basis of the inherent structure information of the given partially observed Petri net. Subsequently, we illustrate the algorithm with the input place of fault transition $\vec{p}_f$. We also apply the reduction rules and calculate all the up-level functions in FDSN $\vec{P}_f$. The tokens flow into FDSN $\vec{P}_f$ and the observable transitions in $\pi_u^{\vec{p}_f}$ are fired. The level function and marking are updated. After the transitions with the new level function are fired, new transitions are once again fired. The level function and marking will stop to update only after the tokens move into $\vec{p}_f$. At this phase, the variation of $M(\vec{p}_f)$ is the maximal number of flow-in of $\vec{p}_f$.

Assume that the number of inputs of $\vec{p}_f$ is $|m|$. Each $\pi_u^{\vec{p}_f}$ has $|p|$ nodes, where $1 \leq q \leq m$. The length of the observed sequence is $|g|$. The complexity of calculating the maximal number of flow-in of place $\vec{p}_f$ is $(\prod_{1 \leq i \leq m} (|p| - 1)) \cdot |m| \cdot |g|$.

Corollary 1. In FDSN $\vec{P}_f$, if $\pi_u^{\vec{p}_f}$ contains an or-split structure, then the maximal number of flow-in of $\vec{p}_f$ (or $\vec{p}_f$) cannot be calculated using Algorithm 1.

Proof. The or-split structure comprises a single place p and multiple outputs. When the number of tokens in place p is known, we cannot know with certainty the exact times in which each output is fired, which means that the number of tokens that flow into other places cannot be calculated. Furthermore, because of the uncertain number of tokens in FDSN $\vec{P}_f$, the maximal number of flow-in of $\vec{p}_f$ cannot be calculated. $\square$

Definition 24. Given an observed sequence ω in a Petri net, if $\exists \omega'' = \rho(\check{T}(p), \omega)$, the minimal number of flow-out of place p with ω is $\delta(p, \omega) = \min[M(p) - M_0(p)]$, $M \in M_{\omega''}$.

Algorithm 2 is used for calculating the minimal number of flow-out of place $\vec{p}_f$ (or $\vec{p}_f$).

Algorithm 2 is proposed on the basis of the inherent structure information of the given partially observed Petri net. Subsequently, we illustrate the algorithm with the input place of fault transition $\vec{p}_f$. With the use of reduction rules, the down-level functions in FDSN $\vec{P}_f$ should be calculated. For any observable transition t belonging to an observed sequence ω'' , if its firing time is less than the number of tokens in its inputs, then $M' = M + C(\cdot, t)$ should be calculated. If the firing time is more than the number of tokens in the inputs, then the unobservable transitions with down-level function $H(t) + 1$ need to be fired to supplement the tokens in the inputs of transition t . By parity of reasoning, the unobservable transitions with highest down-level function, $\vec{p}_f$ is fired to supplement the tokens in the inputs of t . After the observed sequence is fired, the number of supplements is the minimal number of flow-out of place $\vec{p}_f$.

Assume that the number of outputs of $\vec{p}_f$ is $|n|$. Each $\pi_d^{\vec{p}_f}$ has $|q|$ nodes, where $1 \leq q \leq n$. The length of the observed sequence is $|k|$. The complexity of calculating the minimal number of flow-out of place $\vec{p}_f$ is $(\prod_{1 \leq j \leq n} (|q| - 1)) \cdot |n| \cdot |k|$.

```

Input: FDSN  $\vec{P}_f$ , an observed sequence  $\omega$ , initial marking  $M_0$ 
Output:  $\theta(\vec{p}_f, \omega)$ 
(1) for each  $t \in \hat{T}(\vec{p}_f)$  do
(2)   derive  $K(t)$ 
(3) end for
(4) let  $i = 0, M = M_0$ 
(5)  $T_i = \{t \in \hat{T}(\vec{p}_f) \mid K(t) = i\}$ 
(6) let  $T = T_i$ 
(7) if  $T \neq \emptyset$  then
(8)   if transition  $t$  is enabled with the marking  $M$  then
(9)     let  $M = M + C(\cdot, t)$ , go to 8
(10)  else
(11)    let  $T = T - \{t\}$ , go to 7
(12)  end if
(13) else
(14)   let  $i = i + 1$ 
(15) end if
(16) if  $i \leq K(\vec{p}_f)$  then
(17)   go to 5
(18) else
(19)   let  $\theta(\vec{p}_f, \omega) = M(\vec{p}_f) - M_0(\vec{p}_f)$ 
(20) end if
(21) end the algorithm

```

ALGORITHM 1: Computation of the maximal number of flow-in.

Corollary 2. In FDSN $\vec{P}_f$, if $\pi_d^{\vec{p}_f}$ contains an or-joint structure, then the minimal number of flow-out of $\vec{p}_f$ (or $\bar{p}_f$) cannot be calculated using Algorithm 2.

Proof. The or-joint structure comprises a single place p and multiple inputs. When the number of tokens in place p is known, we cannot know with certainty the exact times in which each input is fired, which means that the number of tokens that flow out of other places cannot be calculated. Furthermore, because of the uncertain number of tokens in FDSN $\vec{P}_f$, the minimal number of the flow-out cannot be calculated. $\square$

3.5. Online Fault Diagnosis. In this part, we deal with the problem of specifying a diagnoser that can detect observable transitions regardless of a normal or faulty system behavior.

Definition 25. Given an observed sequence ω in a Petri net N , $\Phi(p, \omega) = \max M(p)$ is called the maximal retention number of place p with ω .

Corollary 3. Given a fault transition t_f and an observed sequence ω in a FDSN $\vec{P}_f$, $\Phi(\vec{p}_f, \omega) = \theta(\vec{p}_f, \omega) - \delta(\vec{p}_f, \omega) + M_0(p)$, M_0 is the initial marking of the reduced model.

Proof. In FDSN $\vec{P}_f$, the initial number of tokens in place $\vec{p}_f$ is $M_0(\vec{p}_f)$ when the reduction rules are applied. After all of the unobservable transitions in $\pi_u^{\vec{p}_f}$ are fired, the maximal number of flow-in is $\theta(\vec{p}_f, \omega) = M(\vec{p}_f) - M_0(\vec{p}_f)$ and the new marking is M . Then, all the unobservable transitions

in $\pi_d^{\vec{p}_f}$ are fired, and the new marking is M' and the minimal number of flow-out is $\delta(\vec{p}_f, \omega) = M(\vec{p}_f) - M'(\vec{p}_f)$. The maximal retention number of $\vec{p}_f$ is

$$\begin{aligned}
 \Phi(\vec{p}_f, \omega) &= M'(\vec{p}_f) \\
 &= \{ [M(\vec{p}_f) - M_0(\vec{p}_f)] \\
 &\quad - M(\vec{p}_f) - M'(\vec{p}_f) \} + M_0(\vec{p}_f) \\
 &= \theta(\vec{p}_f, \omega) - \delta(\vec{p}_f, \omega) + M_0(p). \quad \square
 \end{aligned} \tag{5}$$

Definition 26. A diagnoser is a function $\Delta: T_o^* \times T_f \rightarrow \{0, 1, 2\}$. For each observed sequence ω , the following sets hold:

- (i) For $\Delta(\omega, t_f) = 0$, the behavior of the system is normal during the observed sequence ω
- (ii) For $\Delta(\omega, t_f) = 1$, the behavior is ambiguous
- (iii) For $\Delta(\omega, t_f) = 2$, the behavior of the system is faulty at the observed sequence ω

Theorem 1. Given a fault transition t_f and an observed sequence ω in a Petri net, if there exists a FDSN $\vec{P}_f$ and a FDSN $\bar{P}_f$, and no or-split structure exists in $\pi_u^{\vec{p}_f}$ and $\pi_u^{\bar{p}_f}$, and no or-joint structure exists in $\pi_d^{\vec{p}_f}$ and $\pi_d^{\bar{p}_f}$, then the following are satisfied:

- (i) For existing $\vec{p}_f$, $\Phi(\vec{p}_f, \omega) = 0$, which leads to $\Delta(\omega, t_f) = 0$

Input: FDSN $\vec{p}_f$, an observed sequence ω , initial marking M_0
Output: $\delta(\vec{p}_f, \omega)$

```

(1) for each  $t \in T(\vec{p}_f)$  do
(2)   derive  $H(t)$ 
(3) end for
(4)  $\omega'' = \rho(\vec{T}(\vec{p}_f) | H(t) = i)$ 
(5) let  $T = T_i$ ,  $M = M_0$ ,  $i = 0$ ,  $q = 0$ 
(6) if  $\omega''$  is an empty string then
(7)   let  $\delta(\vec{p}_f, \omega) = q$ 
(8) else
(9)   for  $t \in \omega''$ , and  $t$  is the first transition of  $\omega''$  do
(10)    if  $t$  is enabled at  $M$  then
(11)      let  $M = M + C(\cdot, t)$ 
(12)    else
(13)      if  $i = 1$  then
(14)        go to 23
(15)      else
(16)        let  $i = i - 1$ 
(17)      end if
(18)    end if
(19)  end for
(20) end if
(21) let  $\omega'' = \omega'' - \{t\}$ , go to 6
(22) let  $i = i - 1$ 
(23) if  $i = H(\vec{p}_f^*)$  then
(24)   for  $t \in T$  do
(25)    if  $t$  is enabled at  $M$ ,  $q = q + 1$  then
(26)      let  $M = M + C(\cdot, t)$ , go to 6
(27)    else
(28)      let  $\delta(\vec{p}_f, \omega) = q$ 
(29)    end if
(30)  end for
(31) else
(32)   for  $t \in T$  do
(33)    if  $t$  is enabled at  $M$  then
(34)      let  $M = M + C(\cdot, t)$ , go to 6
(35)    else
(36)      let  $T = T - \{t\}$ 
(37)    end if
(38)  end for
(39) end if
(40) if  $T = \emptyset$  then
(41)   let  $i = i + 1$ , go to 23
(42) else
(43)   for  $t \in T$  do
(44)    if  $t$  is enabled at  $M$  then
(45)      let  $M = M + C(\cdot, t)$ , go to 6
(46)    else
(47)      let  $T = T - \{t\}$ 
(48)    end if
(49)  end for
(50) end if
(51) let  $i = i + 1$ , go to 23
(52) for  $t \in T$  do
(53)   if  $t$  is enabled at  $M$ ,  $q = q + 1$  then
(54)     let  $M = M + C(\cdot, t)$ , go to 6
(55)   else
(56)     let  $\delta(\vec{p}_f, \omega) = q$ 
(57)   end if
(58) end for
(59) end the algorithm

```

ALGORITHM 2: Computation of the minimal number of flow-out.

- (ii) For all $\vec{p}_f$, $\Phi(\vec{p}_f, \omega) \neq 0$, $\Phi(\vec{p}_f, \omega) \geq 0$, which leads to $\Delta(\omega, t_f) = 1$
- (iii) For all $\vec{p}_f$, $\Phi(\vec{p}_f, \omega) < 0$, which leads to $\Delta(\omega, t_f) = 2$

Proof. For a fault transition t_f , when the FDSN $\vec{P}_f$ and the FDSN $\vec{P}_f$ exist, the maximal number of flow-in and the minimal number of flow-out of places $\vec{p}_f$ and $\vec{p}_f$ may be calculated. Then, according to Corollary 3, the corresponding maximal retention numbers are $\Phi(\vec{p}_f, \omega)$ and $\Phi(\vec{p}_f, \omega)$. For all places $\vec{p}_f$, $\Phi(\vec{p}_f, \omega) = 0$ means that the increment of tokens in $\vec{p}_f$ equals the initial number of tokens in $\vec{p}_f$, and no tokens are left in $\vec{p}_f$ with the new marking. The fault transition t_f is therefore not enabled and faults cannot occur. Meanwhile, for all places $\vec{p}_f$, $\Phi(\vec{p}_f, \omega) < 0$ means that the number of flow-out is more than the sum of flow-in and the initial number, but this condition is unreasonable for a Petri net system. Therefore, a fault transition t_f needs to occur to supplement the missing number of tokens in place $\vec{p}_f$. If $\Phi(\vec{p}_f, \omega) \neq 0$ and $\Phi(\vec{p}_f, \omega) \geq 0$, then the fault transition t_f is enabled. Although we cannot decide whether the transition is fired or not, faults may still occur.

When states are increased, the computational complexity of the fault diagnosis becomes high. The computational complexity of calculating the maximal retention number in places $\vec{p}_f$ and $\vec{p}_f$ determines the computational complexity of deriving the diagnoser function Δ . Furthermore, the computational complexity of calculating the maximal number of flow-in and minimal number of flow-out was proposed before. Thus the computational complexity of deriving the function Δ is also a polynomial, which allows the method to meet real-time requirements. $\square$

Example 7. Consider the partially observed Petri net model as described in Figure 1. $T_o = \{t_1, t_2, t_3\}$ and $T_{uo} = \{\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4\}$, and the fault transitions are $T_f = \{\varepsilon_1, \varepsilon_2\}$. For the fault transition 1, its input and output are p_1 and p_4 , and the FDSN $\vec{P}_1$ and FDSN $\vec{P}_4$ can be derived, as shown in Figure 7. After using the modified reduction rule (i), the reduced model can be constructed as shown in Figure 8. $M_0(p_1) = 1$, and $M_0(p_4) = 1$. Given an observed sequence $\omega = t_1 t_2 t_3$, on the basis of Algorithms 1–3, we can derive

$$\begin{aligned}
 \theta(p_1, \omega) &= 1, \\
 \delta(p_1, \omega) &= 0, \\
 \theta(p_4, \omega) &= 0, \\
 \delta(p_4, \omega) &= 2, \\
 \Phi(p_1, \omega) &= 2, \\
 \Phi(p_4, \omega) &= -1, \\
 \Delta(\omega, \varepsilon_1) &= 2.
 \end{aligned} \tag{6}$$

The fault transition ε_1 must occur. In a similar manner, we can derive $\Phi(p_5, \omega) = 0$, $\Phi(p_7, \omega) = 0$, $\Delta(\omega, \varepsilon_2) = 0$, and the fault transition ε_2 must not occur.

FIGURE 7: FDSN $\vec{P}_1$ and FDSN $\vec{P}_4$.

FIGURE 8: The reduced FDSN $\vec{P}_1$ and FDSN $\vec{P}_4$.

4. Example

An intelligent warehouse center is provided as an example. It is used to sort and consolidate the goods in some supply chain systems. The sorting operation is performed with an automated guided vehicle (AGV) as described in Figure 9, which is similar to the conveyor network studied in [4].

The intelligent warehouse center can be described in simplified form as follows: First, the AGV will cross seven areas (for example, A_1 to A_7). In these areas, some different operations are designated before the AGV returns to A_1 . Then, two types of parts A and B will be transferred by the AGV from a loading station (for example, A_1) to the buffers B_1 (Z_3) and B_2 (A_5), respectively. After that, there is a bar code reader named R. It would distinguish between these parts and then transfers the information to the supervisor. Through activating these two switches of sw_1 and sw_2 , the supervisor can direct the parts toward the corresponding buffer. Once the switch sw_1 is activated, a part would move toward B_1 (i.e., a part of type 1 can be detected). Similarly, once the switch sw_2 is activated, a part would move toward B_2 (i.e., a part of type 2 can be detected). Finally, the AGV would move to the other areas when the process of driving a part to B_1 (or B_2) is finished. In the end, it would return to A_1 . When the sensors s_3 , s_5 , and s_7 are contacted, it indicates that the AGV has entered the areas A_3 , A_5 , and A_7 . In this model of the intelligent warehouse center, two different classes of faults are considered. The first one is when a part of type 1 exits the system, it exits through the buffer B_2 but not the buffer B_1 , and then, we say a fault f_1 (T_{12}) occurs. The second one is when the AGV starts a cycle but without any part loading, and then, we say a fault of class f_2 (T_1) occurs.

Input: Petri net system, the fault transition t_f , an observed sequence ω
Output: $\Delta(\omega, t_f)$

```

(1) if FDSN  $\vec{P}_f$  and FDSN  $\overleftarrow{P}_f$  exist then
(2)   for all  $\vec{P}_f$  do
(3)     derive  $\theta(\vec{P}_f, \omega)$  and  $\delta(\vec{P}_f, \omega)$ 
(4)   end for
(5)   for all  $\overleftarrow{P}_f$  do
(6)     derive  $\theta(\overleftarrow{P}_f, \omega)$  and  $\delta(\overleftarrow{P}_f, \omega)$ 
(7)   end for
(8) else
(9)   end the algorithm
(10) end if
(11) derive  $\Phi(\vec{P}_f, \omega)$  and  $\Phi(\overleftarrow{P}_f, \omega)$ 
(12) derive  $\Delta(\omega, t_f)$ 

```

ALGORITHM 3: Online fault diagnosis.

FIGURE 9: The sorting operation of an AGV.

The Petri net system $\langle P, T, M_0 \rangle$ with the initial marking $M_0 = [100000010000]^T$ is used to model the system. The normal behaviors are represented by transitions T_2 to T_{11} as shown in Figure 10 (full and black lines), whereas the faulty behaviors are denoted by the unobservable transitions T_1 and T_{12} , as shown in Figure 10 (dashed and red lines). Unobservable transitions are represented by empty sticks. The significance of transitions and places is listed in Table 1.

With the use of reduction rule (i), the diagnosable subnet of this model can be derived as shown in Figure 11. The input of fault transition f_1 is place P_9 , and the output is P_{10} . The input of fault transition f_2 is place P_1 , and the output is P_2 .

Consider an observed sequence $\omega = T_2 T_4$. In Figure 7, the initial markings are $M_0(P_1) = 1, M_0(P_2) = 0, M_0(P_9) = 0$, and $M_0(P_{10}) = 0$. Transition T_2 is fired once, and $\theta(P_2, \omega) = 0, \delta(P_2, \omega) = 1, \Phi(P_2, \omega) = 1; \theta(P_1, \omega) = 0, \delta(P_1, \omega) = 0$, and $\Phi(P_1, \omega) = 1$. $\Delta(T_2, f_2) = 2$, and thus, the fault f_2 must occur. Moreover, $\theta(P_{10}, \omega) = 0, \delta(P_{10}, \omega) = 0, \Phi(P_{10}, \omega) = 0; \theta(P_9, \omega) = 0, \delta(P_9, \omega) = 0$, and $\Phi(P_9, \omega) = 0$. $\Delta(T_2, f_1) = 0$, and thus, the fault f_1 must not occur.

Consider an observed sequence $\omega = T_2 T_8$. We can derive $\theta(P_2, \omega) = 1, \delta(P_2, \omega) = 1, \Phi(P_2, \omega) = 0; \theta(P_1, \omega) = 0, \delta(P_1, \omega) = 1$, and $\Phi(P_1, \omega) = 0$. $\Delta(T_2, f_2) = 0$, and thus, the

fault f_2 must not occur. In addition, $\theta(P_{10}, \omega) = 0, \delta(P_{10}, \omega) = 0, \Phi(P_{10}, \omega) = 0; \theta(P_9, \omega) = 1, \delta(P_9, \omega) = 0$, and $\Phi(P_9, \omega) = 1$. $\Delta(T_2, f_1) = 1$, and the fault f_1 may occur.

Consider an observed sequence $\omega = T_2 T_8 T_{11}$. We can derive $\theta(P_2, \omega) = 1, \delta(P_2, \omega) = 1, \Phi(P_2, \omega) = 0; \theta(P_1, \omega) = 0, \delta(P_1, \omega) = 1$, and $\Phi(P_1, \omega) = 0$. $\Delta(T_2, f_2) = 0$, and thus, the fault f_2 must not occur. In addition, $\theta(P_{10}, \omega) = 0, \delta(P_{10}, \omega) = 1, \Phi(P_{10}, \omega) = -1; \theta(P_9, \omega) = 1, \delta(P_9, \omega) = 0$, and $\Phi(P_9, \omega) = 1$. $\Delta(T_2, f_1) = 2$, and the fault f_1 must occur.

We simulate some more sequences with Algorithm 3, and partially, details of the situations are listed in Tables 2 and 3.

The structure of the Petri net is much simpler than that of the method using the reachability graph, as shown in Figure 11. The number of state markings (polynomial) is also decreased. By considering another method with basis markings, if the observable string is $T_2 T_4$, then the possible occurring sequences are $T_1 T_2 T_3 T_4, T_1 T_2 T_3 T_4 T_5, T_7 T_1 T_2 T_3 T_4$, and $T_7 T_1 T_2 T_3 T_4 T_5$, the results of which are the same as our method. In the basis marking method, the larger the structure of the Petri net is, the higher the occurring sequences will be, which leads to a more complex computation. This limitation is avoided effectively in the present

FIGURE 10: Partially observed Petri net model of sorting operation with an AGV.

TABLE 1: Nodes of partially observed Petri net model of sorting operation with an AGV.

P/T	Significance
$P_1 - P_7$	AGV in area $A_1 - A_7$
P_8	AGV is available to take a new part
$P_9 - P_{10}$	Switch sw_1 (resp., sw_2) is activated
$P_{11} - P_{12}$	Output buffers $B_1 - B_2$
$T_2 - T_7$	Operations in area $A_2 - A_7$ are over
$T_8 - T_9$	AGV exits the loading station with a part of type 1 (resp., 2)
$T_{10} - T_{11}$	A part exits the system and is stored in buffer B_1 (resp., B_2)
T_1	Fault f_2 : AGV exits the loading station without any part
T_{12}	Fault f_1 : a part of type 1 activates the switch sw_2

TABLE 2: Fault situations with an observed sequence ω .

ω	$\Phi(P_9, \omega)$	$\Phi(P_{10}, \omega)$	$\Delta(\omega, f_1)$
T_2	0	0	0
$T_2 T_8$	1	0	1
$T_2 T_8 T_{11}$	1	-1	2
$T_2 T_8 T_{11} T_6$	1	-1	2
$T_2 T_8 T_{11} T_6 T_8$	2	-1	2

TABLE 3: Fault situations with an observed sequence ω .

ω	$\Phi(P_2, \omega)$	$\Phi(P_1, \omega)$	$\Delta(\omega, f_2)$
T_2	-1	1	2
$T_2 T_8$	0	0	0
$T_2 T_8 T_{11}$	0	0	0
$T_2 T_8 T_{11} T_6$	0	1	1
$T_2 T_8 T_{11} T_6 T_8$	1	0	0

FIGURE 11: The reduced model of sorting operation with an AGV.

work because not all of the possible occurring sequences are collected; i.e., we only focus on the inputs and outputs of the fault transitions and calculate the number of tokens that flow in or out. Consequently, we can diagnose the faulty behavior in the polynomial level.

5. Conclusions and Future Work

This work addresses the problem of fault diagnosis of DESs and proposes an online diagnoser on the basis of the partially observed Petri net. The online computation is formulated on a net structure and applied with reduction rules. The method involves an observed sequence of system events, which then

is used as basis for deciding online whether the system behavior is normal, faulty, or uncertain. To achieve the goals, the maximal number of flow-in and the minimal number of flow-out are introduced. By calculating the maximal and minimal numbers for some reduced subnets, the maximal retention number of places $\vec{p}_f$ and $\bar{p}_f$ can be obtained, which means that the number of tokens in places $\vec{p}_f$ and $\bar{p}_f$ can be determined after an observed sequence is fired. Subsequently, the fault transition is enabled (or not). The entire process is formulated using the appropriate reduced subnets with fault information, and it permits to reduce the computational effort to solve the fault diagnosis problem at the price of a small memory increase, which meets the real-time requirement.

Several directions are considered in our future work. First, we want to explore additional reduction rules for the fault diagnosis problem of DESs. In such a case, the structure of the Petri net can be further elaborated. In addition, we plan to explain in detail the relationship between transitions and places given the above special structure. Finally, we will likely extend the proposed method to labeled Petri nets in the nondeterminism context.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] V. Chandra and R. Kumar, "A discrete event systems modeling formalism based on event occurrence rules and precedences," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 785–794, 2001.
- [2] S. Jiang, R. Kumar, and H. E. Garcia, "Diagnosis of repeated/intermittent failures in discrete event systems," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 2, pp. 310–323, 2003.
- [3] X. Xie, "Fluid-stochastic-event graphs for evaluation and optimization of discrete-event systems with failures," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 3, pp. 360–367, 2002.
- [4] D. Lefebvre, "Fault diagnosis and prognosis with partially observed petri nets," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 10, pp. 1413–1424, 2014.
- [5] G. G. Rigatos, "Fault detection and isolation based on fuzzy automata," *Information Sciences*, vol. 179, no. 12, pp. 1893–1902, 2009.
- [6] 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, 2010.
- [7] C. H. Kuo and H. P. Huang, "Failure modeling and process monitoring for flexible manufacturing systems using colored timed petri nets," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 3, pp. 301–312, 2000.
- [8] D. Lefebvre and C. Delherm, "Diagnosis of des with petri net models," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 1, pp. 114–118, 2007.
- [9] O. Contant, S. Lafortune, and D. Teneketzis, "Diagnosability of discrete event systems with modular structure," *Discrete Event Dynamic Systems*, vol. 16, no. 1, pp. 9–37, 2006.
- [10] S. H. Zad, R. H. Kwong, and W. M. Wonham, "Fault diagnosis in discrete-event systems: framework and model reduction," *IEEE Transactions on Automatic Control*, vol. 48, no. 7, pp. 1199–1212, 2003.
- [11] M. P. Cabasino, A. Giua, S. Lafortune, and C. Seatzu, "A new approach for diagnosability analysis of petri nets using verifier nets," *IEEE Transactions on Automatic Control*, vol. 57, no. 12, pp. 3104–3117, 2012.
- [12] M. Dotoli, M. P. Fanti, A. M. Mangini, and W. Ukovich, "On-line fault detection in discrete event systems by petri nets and integer linear programming," *Automatica*, vol. 45, no. 11, pp. 2665–2672, 2009.
- [13] A. Ramirez-Trevino, E. Ruiz-Beltran, I. Rivera-Rangel, and E. Lopez-Mellado, "Online fault diagnosis of discrete event systems. a petri net-based approach," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 1, pp. 31–39, 2007.
- [14] Y. Wu and C. N. Hadjicostis, "Algebraic approaches for fault identification in discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 50, no. 12, pp. 2048–2055, 2005.
- [15] M. Alcaraz-Mejia, E. Lopez-Mellado, A. Ramirez-Trevino, and I. Rivera-Rangel, "Petri net based fault diagnosis of discrete event systems," in *Proceedings of the IEEE International Conference on Systems*, Washington, DC, USA, October 2003.
- [16] D. Lefebvre, "Firing sequences estimation in vector space over Z_3 for ordinary petri nets," *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, vol. 38, no. 6, pp. 1325–1336, 2008.
- [17] C. N. Hadjicostis and G. C. Verghese, "Structured redundancy for fault tolerance in state-space models and petri nets," *Kybernetika—Praha*, vol. 1, no. 1, pp. 39–55, 1999.
- [18] A. Giua and F. Dicesare, "Petri net structural analysis for supervisory control," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 2, pp. 185–195, 1994.
- [19] A. Giua and C. Seatzu, "Fault detection for discrete event systems using petri nets with unobservable transitions," in *Proceedings of the IEEE Conference on Decision and Control*, Seville, Spain, December 2010.
- [20] M. P. Cabasino, A. Giua, and C. Seatzu, "Diagnosability of bounded petri nets," in *Proceedings of the IEEE Conference on Decision and Control*, Shanghai, China, December 2009.
- [21] F. Basile, P. Chiacchio, and G. De Tommasi, "On-diagnosability of Petri nets via integer linear programming," *Automatica*, vol. 48, no. 9, pp. 2047–2058, 2012.
- [22] B. Liu, M. Ghazel, and A. Toguyni, "Toward an efficient approach for diagnosability analysis of des modeled by labeled petri nets," in *Proceedings of the European Control Conference*, Strasbourg, France, June 2014.
- [23] L. Ben, M. Khelif-Bouassida, and A. Toguyni, "On-the-fly diagnosability analysis of labeled petri nets using t-invariants," *IFAC Papersonline*, vol. 48, no. 7, pp. 64–70, 2015.
- [24] C. N. Hadjicostis and G. C. Verghese, "Monitoring discrete event systems using petri net embeddings," in *Proceedings of the International Conference on Application and Theory of Petri Nets*, pp. 188–207, Bonn, Germany, June 1999.
- [25] J. Prock, *A New Technique for Fault Detection Using Petri Nets*, 1991.
- [26] Y. Ru, C. N. Hadjicostis, and N. Christoforos, "Fault diagnosis in discrete event systems modeled by partially observed petri

- nets,” *Discrete Event Dynamic Systems*, vol. 19, no. 4, pp. 551–575, 2009.
- [27] E. Fabre, A. Benveniste, C. Jard, and M. Smith, “Diagnosis of asynchronous discrete event systems: a net unfolding approach,” *IEEE Transactions on Automatic Control*, vol. 48, no. 5, pp. 714–727, 2003.
- [28] T. Murata, “Petri nets: properties, analysis and applications,” *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989.
- [29] F. Basile, P. Chiacchio, and G. De Tommasi, “An efficient approach for online diagnosis of discrete event systems,” *IEEE Transactions on Automatic Control*, vol. 54, no. 4, pp. 748–759, 2009.
- [30] J. L. Luo, W. M. Wu, H. Y. Su, and J. Chu, “Supervisor synthesis for enforcing a class of generalized mutual exclusion constraints on petri nets,” *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, vol. 39, no. 6, pp. 1237–1246, 2009.
- [31] L. X. Zhou, H. G. Ying, and K. S. Ho, “Applying petri-net-based reduction approach for verifying the correctness of workflow models,” *Wuhan University Journal of Natural Sciences*, vol. 11, no. 1, pp. 203–210, 2006.
- [32] M. L. Ben, Khelif-Bouassida, and A. Toguyeni, “Diagnosis and diagnosability analysis of labeled petri nets using reduction rules,” in *Proceedings of the International Workshop on Discrete Event Systems*, Xi’an, China, June 2016.