Petri-Net Based Approach To Configure Online Fault Diagnosis Systems for Batch Processes

Yi-Chung Chen, Ming-Li Yeh, Chia-Lun Hong, and Chuei-Tin Chang*

Department of Chemical Engineering National Cheng Kung University Tainan, Taiwan 70101, Republic of China

Online fault diagnosis is a task of critical importance for maintaining a high level of operational safety in many chemical plants. The Petri-net models are adopted in this work for describing the fault propagation behaviors in batch processes. A systematic method has been developed to synthesize a timed Petri-net hierarchically structured according to any given piping and instrumentation diagram (P&ID) and its operating procedure. On the basis of this model, a diagnoser can be constructed automatically with a computer program for online implementation. Computer algorithms have also been devised to place additional sensors and/or synthesize extra operation steps for the purpose of improving diagnostic performance. Several examples are presented in this paper to demonstrate the effectiveness and correctness of the proposed approach.

Introduction

Unexpected catastrophic events such as fires, explosions, or toxic releases may occur in the course of manufacturing chemicals in every process plant. A serious accident may cause not only casualties and property losses but also serious damages to the ecosystem. Online fault diagnosis has always been considered as an effective means for enhancing operational safety. However, most available methods developed so far are for the *continuous* chemical processes. These approaches could be classified into three distinct groups,[1−3] i.e., the model-based approaches, the knowledge-based approaches, and the data-analysis-based approaches. Due to unsteady operating conditions, application of these approaches to batch chemical processes is usually difficult, and therefore, significantly fewer works have been reported in the literature. On the basis of statistical analysis, Nomikos and MacGregor[4,5] developed a systematic methodology for batch process monitoring, which has also been adopted for online fault diagnosis.[6−9] Moreover, fault identification strategies based on a combination of artificial neural networks and knowledge-based expert systems[10,11] and the observer techniques[12] have also been developed for the batch operations.

It can be observed from the aforementioned studies that, in order to facilitate effective diagnosis, the fault propagation mechanisms must be characterized with a model and the symptom evolution patterns caused by every fault origin must also be *predicted* in advance. Notice that, although the digraph model is by far the most popular choice for this purpose,[13,14] it is useful mostly in applications associated with the continuous processes. This is because of the fact that the digraph is not suitable for describing the dynamic causal relationships among events, time, equipment states, and system configurations in the semibatch or batch manufacturing processes. Since Petri-net is well-known for its capability in characterizing discrete event systems,[15,16] it is adopted in this work as a modeling tool to circumvent the above drawbacks.

Failure diagnosis in the discrete-event systems (DESs) was first studied in the work of Sampath et al.[17,18] The notion of system diagnosability was defined by these authors and, in addition, a systematic procedure was proposed to construct the diagnoser for a given DES. A diagnoser was constructed to serve two main purposes in the above studies. In addition to its obvious capability in online fault diagnosis, it is also useful for verifying diagnosability. Similar studies have also been performed by a number of other research groups.[19−24] Sampath's study was also extended by Ushio et al.[25] with Petri-net models to better describe the discrete-event systems. Only a portion of the equipment states, i.e., the token numbers in places, were assumed in such models to be observable, while the events, i.e., the firings of transitions, were considered to be completely undetectable. By treating all hardware failures as unobservable events, Jiang et al.[26] handled the diagnosis problems with automatic modeling techniques. An online diagnosis approach was also developed in the work of Jiroveanu et al.[27,28] on the basis of timed Petri-nets. Chung[29] recently modified the diagnoser generation algorithm by allowing the transitions to be partially observable. The above studies mostly focused on checking diagnosability, while ignoring the practical issue of resolution enhancement. Notice also that the proposed models were constructed by using the ordinary Petri-net (without time-delayed transitions, inhibitor arcs, and test arcs), and thus, the applicability of the resulting diagnosers in realistic cases is limited.

Since the scopes of the aforementioned existing publications on DES were mainly concerned with mechanical and/or electrical systems, a diagnoser-building algorithm is developed in the present work for applications in the *batch chemical processes*. To facilitate implementation efficiency, this algorithm has been encoded in a commercially available programming software Delphi 2007,[30] which is one of the most respected and widely used rapid application development (RAD) environments today, to generate the diagnoser tree automatically according to a *timed* Petri-net model. As mentioned previously, a diagnoser constructed with the given process configuration and operating procedure may not be able to uniquely identify all possible fault origins. Systematic procedures have thus been developed in this work to identify diagnosable scenarios and to evaluate resolution level of the diagnostic system. Two resolution enhancement strategies have been proposed in this study. The first approach is to place additional sensors, while the second is to execute additional operation steps which are not included in the original

* To whom correspondence should be addressed. Tel.: 886-6-2757575, ext. 62663. Fax: 886-6-2344496. E-mail: ctchang@mail.ncku.edu.tw.

10.1021/ie901410p
Published on Web 03/29/2010

recipe. Both strategies can be synthesized automatically according to the Petri-net model in a straightforward fashion.

Petri-Net Representations of Batch Operations

In this study, the Petri-net based diagnosers are used for performing online fault identification in batch process systems, for checking diagnosability and for assessing diagnostic resolution level. A generalized Petri-net representation of the batch processes has been developed for these purposes. The detailed model configuration is outlined in the sequel:

Component Models. Let us first consider the 6-tuple framework of a timed Petri-net:

$$\mathscr{G} = (\mathbb{P}, \mathbb{T}, \mathbb{F}, \mathbb{W}, \mathbf{m}_0, \mathbf{t}_d) \tag{1}$$

where, $\mathbb{P} = \{p_1, p_2, ..., p_n\}$ and $\mathbb{T} = \{t_1, t_2, ..., t_m\}$ represent, respectively, the sets of places and transitions in the Petri-net; $\mathbb{F}$ is the union of the sets of place-to-transition and transition-to-place arcs, i.e., $\mathbb{F} \subseteq (\mathbb{P} \times \mathbb{T}) \cup (\mathbb{T} \times \mathbb{P})$; $\mathbb{W}$ denotes the set of weighting functions associated with the arcs in $\mathbb{F}$; $\mathbf{m}_0$ is the vector in which the initial token number in every place is stored; $\mathbf{t}_d$ is the vector in which the delay times of all transitions are included. It should be noted that the arcs in a Petri-net can be classified into three different types: (1) *normal arc* is that represented by a directed solid line; (2) *test arc* is that represented by a directed dash line; (3) *inhibitor arc* is that represented by a directed solid line with a small circle at its end. Notice also that, when a normal arc is connected to a particular transition, a set of tokens in its input place (whose number equals the weight on the place-to-transition arc) should be removed after this transition is fired and another set of tokens (whose number equals the weight on the transition-to-place arc) should be deposited into the output place. On the other hand, tokens should *not* be removed after firing when an inhibitor arc or a static test arc is connected. A test arc is equivalent to two equally weighted normal arcs in opposite directions, while an inhibitor arc is used in modeling the failure mechanisms that inhibit certain normal events.

Every hardware item in a batch process can be viewed as a component and modeled with a timed Petri-net. Generally speaking, two different types of components can be identified according to their states. If a finite number of definite equipment states can be clearly identified, then each state should be represented with a place and the transition from one state to another is usually instantaneous. For example, let us consider the valve model presented in Figure 1. In this model, places p_1 (V1C) and p_2 (V1O) are used to represent the close and open positions respectively, while the *untimed* transitions t_1 and t_2 denote the corresponding close-to-open and open-to-close actions. Notice that additional places may have to be added to specify other preconditions for triggering or inhibiting these two events. For example, if the valve position is adjusted remotely with a programmable logic controller (PLC), then the instrument states causing the operation steps must also be modeled. These states can be represented with the input places of t_1 and t_2 (i.e., p_{14} and p_{13}), respectively. As another example, let us consider the failed valve states represented by p_3 (V1SC), i.e., V-1 sticks at the close position, and p_4 (V1SO), i.e., V-1 sticks at the open position. The output inhibitor arcs from these places are used in this model to prevent firing of the corresponding transitions.

Notice that the token number in each place of the proposed Petri-net model can only assume nonnegative integer values. If the component state varies continuously with time, e.g. level, temperature, or pressure, then an approximated model should be constructed to qualitatively characterize the time profile of this state. Specifically, a finite number of discretized states must be identified and, also, a nonzero delay time must be selected to represent the elapsed time of each state transition process. For example, let us consider the component model of a storage tank (see Figure 2). Without loss of generality, let us assume that only two discretized states are important, i.e., empty (TL) and full (TH), and the transition times between these two states both equal one time unit. The preconditions for triggering or inhibiting the empty-to-full and full-to-empty transitions are again the equipment states of other components, i.e., flow in outlet pipeline (OPF) and flow in inlet pipeline (IPF).

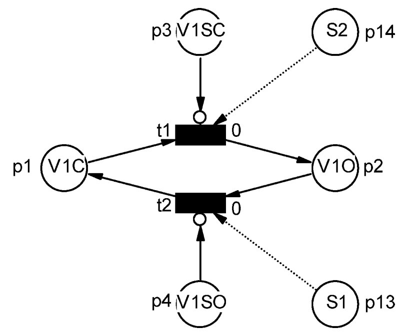

Figure 1. Discharge valve model (V-1).

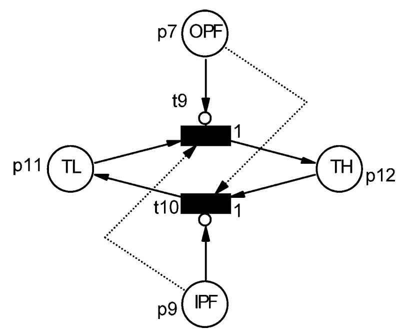

Figure 2. Tank model.

System Model. Any batch process can be fully described with a piping and instrumentation diagram (P&ID) and a sequential function chart (SFC). The corresponding system model can be assembled with its component models. As mentioned before, every hardware item in the P&ID (including the controller) should be treated as a component. Since the preconditions for triggering or inhibiting the transitions in every such model are usually the states of other components, the system model can be obtained simply by collecting a complete set of component models.

To illustrate this model-building approach, let us consider the liquid storage system shown in Figure 3, which will be referred to as example 1 later in this paper. The height of liquid level in this tank is monitored online. Two distinct sensor signals, i.e., (1) LH (level high) and (2) LL (level low), are sent to a PLC to actuate the control valves (V-1 and V-2) on the outlet and inlet pipelines respectively. In response to the LH signal, V-1 is opened while V-2 closed. On the other hand, LL signal triggers the control actions to close V-1 and to open V-2. It is assumed that this operation is periodical and the above two sets of control actions are repeated in every period. Under the assumptions that the initial liquid level in the tank is low and V-1 and V-2 are at the close position initially, a sequential function chart can be constructed to represent the needed cyclic operating

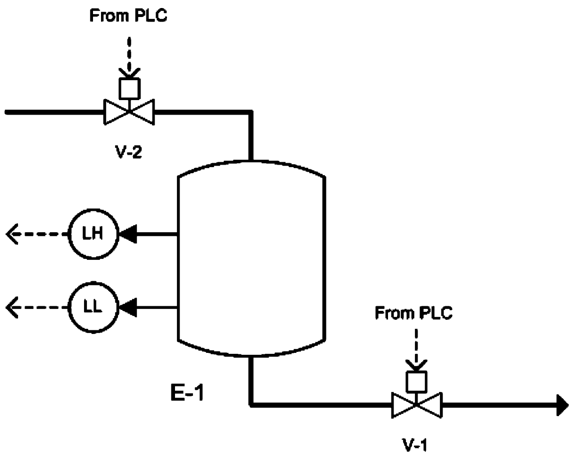

Figure 3. Simple liquid storage system (example 1).

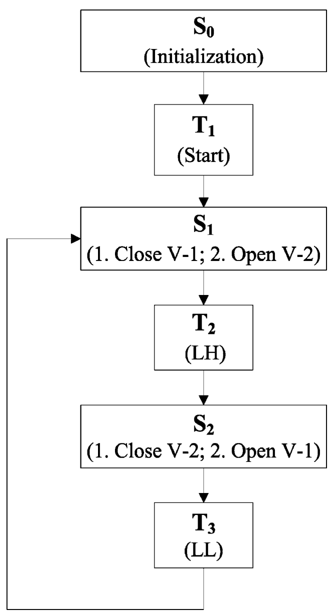

Figure 4. Sequential function chart of the operating procedure in example 1.

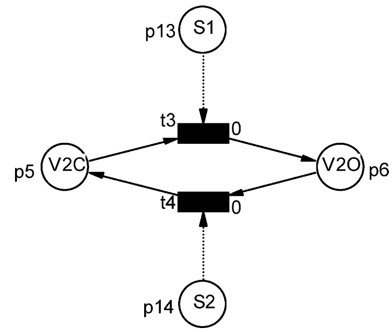

Figure 5. Inlet valve model (V-2).

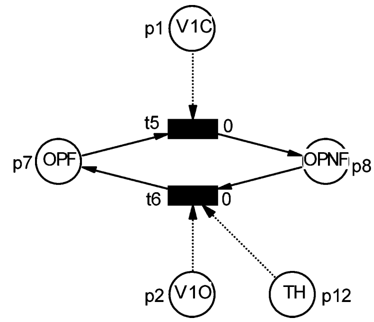

Figure 6. Outlet pipeline model.

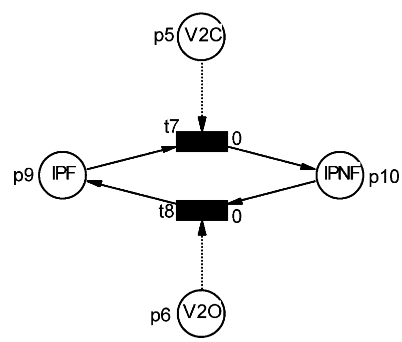

Figure 7. Inlet pipeline model.

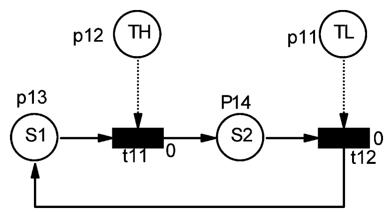

Figure 8. Controller model.

procedure (see Figure 4). Notice that S_i ($i = 0, 1, 2$) and T_j ($j = 1, 2, 3$) denote the operation steps and the activation conditions of these steps, respectively. Notice also that the control actions taken in each step, and the sensor signals used in each condition are also specified in this chart.

According to the aforementioned conventions, the components in example 1 should be: the two valves (V-1 and V-2), the inlet and outlet pipelines, the tanks, and the PLC controller. The Petri-net representations of V-1 and tank have already been presented in Figures 1 and 2, respectively. The remaining component models can be built according to the same principles and, for the sake of brevity, they are included in Figures 5−8 without further explanation. Since these Petri-nets are coupled, they can be directly connected to form a system model of 14 places, 12 transitions, and 41 arcs.

Failure Models. After building the above models to represent normal system behaviors, additional mechanisms should be incorporated into each component Petri-net to characterize the effects of failures. The general structure in Figure 9 is adopted in the present study to represent *all* possible fault scenarios. In this model, the direct outcome of a failure is viewed as a change in the equipment state of the failed component. The state caused by the ith failure mode is represented by the place PFS(i) (i = 1, 2, ...). The effects of a failure are regarded as the outcomes created by replacing a set of routine events occurred during normal operation with an alternative set of abnormal events. These effects can be readily modeled with a combination of the inhibitor arcs and test arcs (see Figure 9). The former arcs are used to disable the transitions corresponding to the routine (normal) events, i.e. TN(j) (j = 1, 2, ..., m), and the latter activate the alternative transitions representing the failure events, i.e. TF(k) (k = 1, 2, ..., n).

Let us first use the Petri-net given in Figure 1 as an example to illustrate this model-building approach. In particular, the abnormal valve states, i.e., “V-1 sticks at the close position” (V1SC) and “V-1 sticks at the open position” (V1SO), are

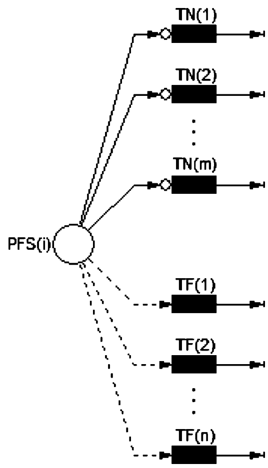

Figure 9. General failure model.

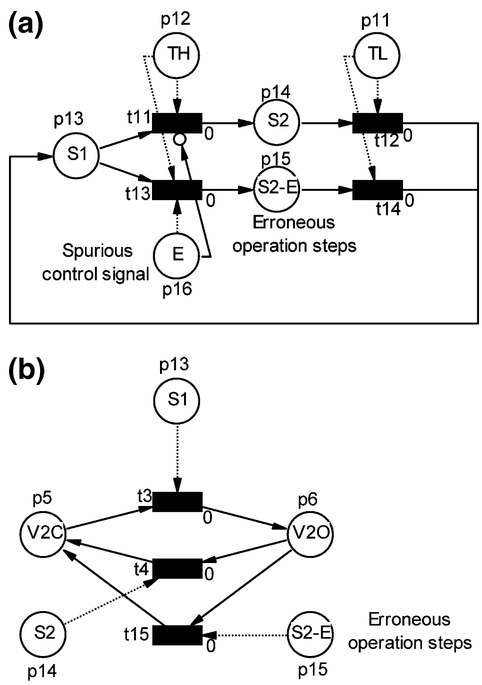

Figure 10. (a) Controller model that contains an additional failure module. (b) Inlet valve model that contains an additional failure module.

represented with p_3 and p_4, and their effects are characterized with inhibitor arcs only. Notice that no test arcs are needed in this case.

Let us next assume that the PLC may occasionally send out spurious signals to execute erroneous operation steps. If an additional fault origin, i.e., the spurious control signal, is to be considered in the operation of V-2, then the controller model in Figure 8 and the valve model in Figure 5 could be changed to those shown in Figure 10a and b, respectively.

To address the issue of state explosion, it is also assumed in this work that a failure event can only occur immediately before the affected observable event. For instance,

- The failure event "V1SC" may occur just before the action "open V-1" is executed.
- The failure event "tank leak" may occur while the liquid level in tank is about to change.

Classification of Places and Transitions. It is assumed in this work that not all equipment states can be monitored online. In other words, the places in $\mathbb{P}$ can be classified as observable and unobservable and then collected in two corresponding subsets, i.e., $\mathbb{P} = \mathbb{P}_o \cup \mathbb{P}_u$. Notice that some of the unobservable states may be caused by failures. Thus, there is a need to further distinguish the unobservable normal and failed states, i.e., $\mathbb{P}_u = \mathbb{P}_n \cup \mathbb{P}_f$.

On the other hand, the transitions in the Petri-net model can also be divided into two groups to represent the normal and abnormal events respectively, i.e., $\mathbb{T} = \mathbb{T}_N \cup \mathbb{T}_F$. The events represented by the elements in $\mathbb{T}_N$ are associated with normal state-transition processes, while those in $\mathbb{T}_F$ can be considered as failure events. It is assumed in this study that almost all events occur instantaneously except for some in the former case. In other words, some transitions in $\mathbb{T}_N$ may be fired after finite time delays to better characterize the realistic system behavior. Notice also that, in addition to the transitions in $\mathbb{T}_F$, the places in $\mathbb{P}_f$ may have to be linked to some of the transitions in $\mathbb{T}_N$ with inhibitor arcs to model the failure effects. Thus, the normal transitions can be further classified as $\mathbb{T}_N = \mathbb{T}_A \cup \mathbb{T}_B$. Specifically, $\mathbb{T}_A$ represents the subset of transitions which are unaffected by such failures, while $\mathbb{T}_B$ is the subset of affected ones. Finally, it should be noted that, other than the online measurements, controller execution of a specific operation step is considered in this study as a *known* event also available for diagnosis. All transitions in the Petri-net model can therefore be classified according to this alternative criterion, i.e., $\mathbb{T} = \mathbb{T}_K \cup \mathbb{T}_{UK}$, where $\mathbb{T}_K$ denotes the set of transitions representing the controller actions, and $\mathbb{T}_{UK}$ is the set of remaining transitions. It should be noted that the former transitions (in $\mathbb{T}_K$) may either be in $\mathbb{T}_N$ or in $\mathbb{T}_F$.

For illustration simplicity, let us assume in the aforementioned batch process (i.e., Figures 3 and 4) that V-1 is the only component that might fail. As a result, the places and transitions in the component Petri-nets (i.e., Figures 1, 2, and 5−8) can be classified as follows:

$$\mathbb{P}_o = \{p_{11}, p_{12}\} \tag{2}$$

$$\mathbb{P}_n = \{p_1, p_2, p_5, p_6, p_7, p_8, p_9, p_{10}, p_{13}, p_{14}\} \tag{3}$$

$$\mathbb{P}_f = \{p_3, p_4\} \tag{4}$$

$$\mathbb{T}_A = \{t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}\} \tag{7}$$

$$\mathbb{T}_B = \{t_1, t_2\} \tag{8}$$

$$\mathbb{T}_F = \varnothing \tag{9}$$

$$\mathbb{T}_K = \{t_{11}, t_{12}\} \tag{10}$$

$$\mathbb{T}_{UK} = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}\} \tag{11}$$

Diagnoser Construction Procedure

If fault diagnosis is to be performed online, a connection between the observable symptoms and their root cause(s) must be established as quickly as possible. Since a correct judgment can only be made with sufficient information, it may not be possible to uniquely identify every fault origin with the available sensors. Therefore, it is very important to check the diagnosability of each root cause in advance.

Although all possible fault propagation scenarios in a given system can be predicted (or simulated) with the aforementioned system model, it is not convenient to use Petri-net directly for online diagnosis and for offline evaluation of diagnostic performance. To facilitate efficient implementation, the Petri-

Table 1. Information Stored in a Diagnoser Node

symbol	description
NodeNumber	a distinct label assigned to the present system state
CurrentMarking	a vector containing the current token numbers in all places
CurrentTime	a time stamp denoting the current time
TransitionFired	the labels of transitions previously red to produce the present marking
ParentNode	the label of parent node (from which the present node is originated)
ChildNodes	the labels of child nodes (which can be generated from the present node)
ChildNodeCount	the total number of child nodes
FailureRecord	a record of all failures that have already occurred until the current time

net model is transformed into a "diagnoser" in this work for enumerating all possible system states under normal/abnormal operating conditions and also the state evolution sequences. This diagnoser is structured as a tree with nodes denoting the possible states. A data set is constructed for each node to incorporate the needed information and the detailed descriptions of these data can be found in Table 1. It is assumed that the initial system state is given and the corresponding Petri-net marking is stored in the data set embedded in the root node of diagnoser tree. The remaining states in the tree can be identified by firing the enabled transitions sequentially according to the system model.

Enumeration of Diagnoser Nodes. As an example of the enumeration procedure, let us again consider the simple storage system presented in Figure 3, the operating procedure in Figure 4, and the corresponding Petri-net models in Figures 1, 2, and 5−8. Every possible system state can be represented with a *marking* of the Petri-net model, i.e., a row vector in which the token numbers of all places are sequentially stored. The initial marking in this case is (10001001011010), i.e., the tank is empty and valves 1 and 2 are at the close position. Notice that the underlined two digits in this marking denote the observable tank states, i.e., "empty" (p_{11}) and "full" (p_{12}).

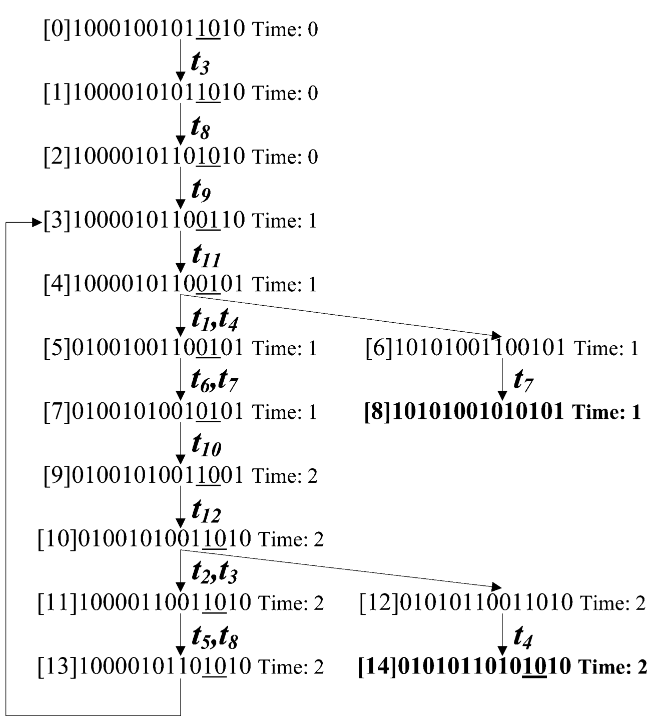

Figure 11. Diagnoser tree for example 1.

Figure 11 is the fully developed diagnoser tree for the simple example mentioned above. The root node of this tree (node [0]) is characterized with the initial marking, while the other nodes can be enumerated by firing the enabled transitions sequentially. Specifically, since a token is in p_{11}, transition t_{12} should be enabled initially (see Figure 8). As a result, the PLC will try to implement operation step S_1 in Figure 4, i.e., (1) close V-1 and (2) open V-2, after the operation begins. The former step cannot enable t_2 since V-1 is already closed. However, the latter step fires transition t_3 and the resulting marking is (10000101011010), i.e., node [2]. Having fired all transitions which are affected by the operation steps in S_1, the enabled transitions should then be processed next. From Figure 7, it is clear that t_8 can be triggered since a token is present in each of its two input places, i.e., p_6 and p_{10}. The resulting marking is (10000101101010), which is stored in node [2] of the diagnoser. Since the conditions associated with p_9 and p_{11} in Figure 2 now become valid, transition t_9 should then be fired to yield node [3] with the marking (10000101100110). Notice that this state can only appear after 1 unit of time is elapsed due to the time delay assigned to t_9. At this point, it can be observed that none of the transitions in the Petri-net model can be enabled without executing additional operation steps in SFC.

Notice that the marking in node [3] satisfies the activation condition T_2 (in Figure 4) for issuing the commands in S_2 and thus transition t_{11} should be fired. The resulting marking is given in node 4, i.e., (10000101100101). A token in p_{14} triggers two events, i.e., (1) close V-2 and (2) open V-1, which are represented with transitions t_4 (in Figure 5) and t_1 (in Figure 1), respectively. These two transitions can both be fired to produce the marking (01001001100101) in node [5]. It should be noted that, other than p_1, the firing of t_1 is also dependent upon the state of p_3, which is associated with the failure "V-1 sticks at the close position" (V1SC). There are thus two possibilities at the instance when the step "open V-1" is carried out. If the valve is in normal condition, transition t_1 (and also t_4) can be fired to generate the marking in node [5]. However, if the aforementioned failure is present, t_1 is inhibited by the condition in p_3 and the corresponding system state can be described with the marking (10101001100101) in node [6]. Consequently, the diagnoser is branched at the point when the second operation step in S_2 is performed. The marking in node [5] can trigger t_6 in Figure 6 (caused by the states given in p_2, p_8, and p_{12}) and t_7 in Figure 7 (caused by the conditions in p_5 and p_9) to produce the marking (01001010010101) in nodes [7], which in turn triggers t_{10} in Figure 2 to produce (01001010011001) in node [9]. Since a time delay of 1 unit is needed in firing the latter transition, the state in node [9] can be reached only after time 2. On the other hand, the marking in node [6] can fire t_7 in Figure 7 (due to the states of p_5 and p_9) to yield the marking (10101001010101) in node [8].

It is obvious that the diagnoser tree can be developed further with the aforementioned enumeration method. However, since the operation specified in Figure 4 is cyclic in nature, an infinitely large tree will be produced if this approach is used alone. A proper diagnoser construction procedure should therefore also include the propagation and termination functions described below.

Propagation and Termination Functions. As mentioned before, a batch process can be fully described with a P&ID and the corresponding SFC. It is assumed in this study that the operating procedure in SFC is properly defined in such a way that each system state can be driven to one and only one other

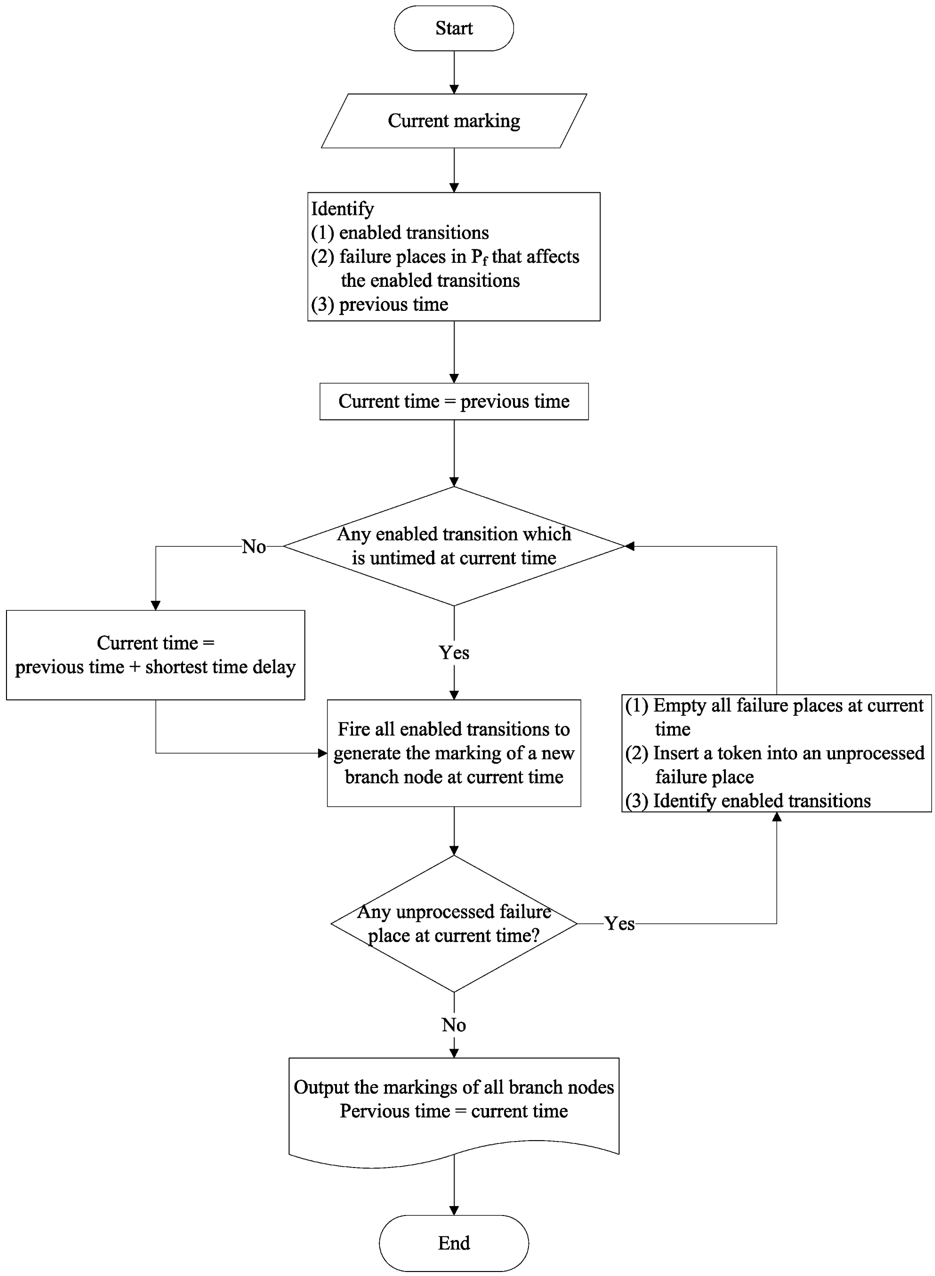

Figure 12. Flowchart of propagation function.

state with a controller under normal conditions. Consequently, the split branches in a diagnoser tree are *always* caused by failures.

A propagation function has been developed in this work to generate the branched child nodes from a given parent node. The input of this function is the current marking, while the outputs are the data sets associated with all the child nodes. The detailed propagating procedure is summarized in Figure 12. Basically the standard rules for firing transitions are followed to facilitate propagation in normal conditions and also after failure occurs. Notice that the implied assumption of this procedure is that only a single-fault scenario is possible at any instance. This assumption is justifiable since the probability of multiple faults occurring simultaneously should be extremely low.

If the above propagation procedure is carried out indefinitely, an extremely large tree may be created. This tree can be reduced to a manageable size by terminating the tree-building process at the repeated nodes. A flowchart of the termination procedure is shown in Figure 13. This termination check is performed on a candidate node immediately after a propagation step (see Figure 14). Generally speaking, a diagnoser tree branch is terminated if (1) an infinite loop can be detected or (2) no enabled transitions are present. To check the former criterion, the data set of current node should be compared with those of all its preceding nodes. If an exact match is identified, then the current node can be considered as terminal. Note that, in this case, the current node and the matched candidate must share the same set of child nodes.

The flowcharts presented in Figures 12−14 have been encoded in a generic computer program for automatically generating the diagnoser trees based on given Petri-net models. As an example, the completed tree in Figure 11 was produced with the proposed approach. A copy of this code can be found in the Supporting Information—part I. It should be noted that the diagnoser of a given system should be unique.

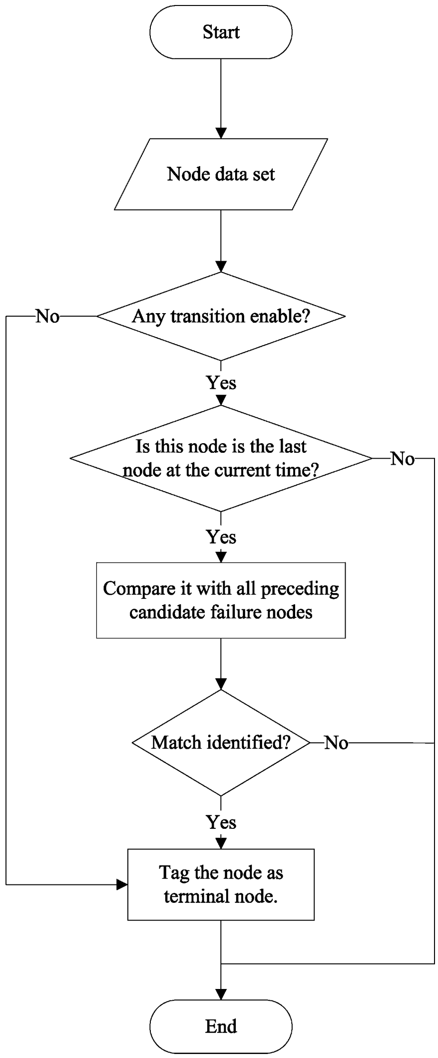

Figure 13. Flowchart of termination function.

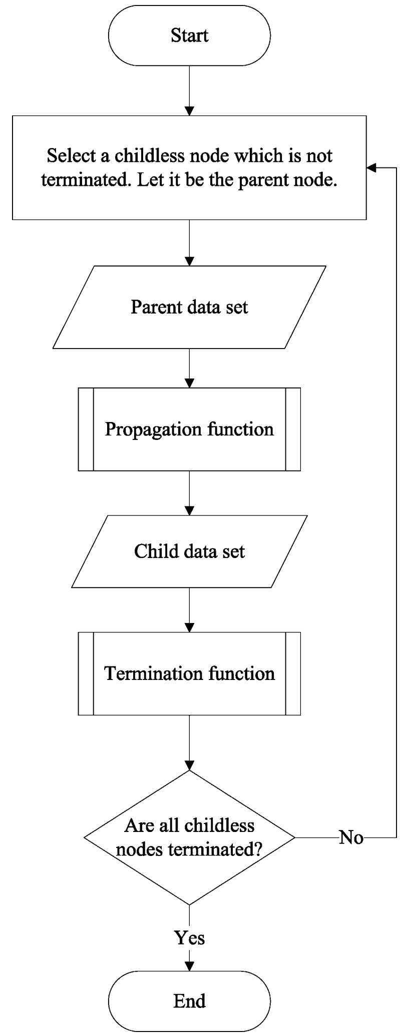

Figure 14. Flowchart for node enumeration procedure.

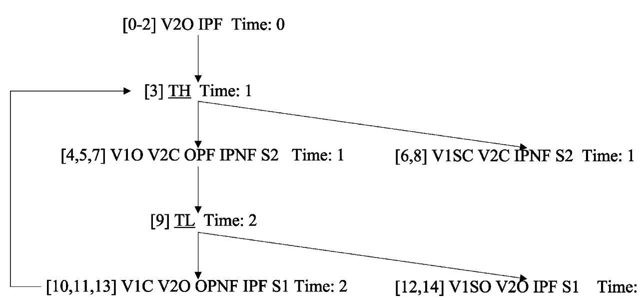

Figure 15. State-time tree of example 1.

State-Time Tree. For convenience, the fully developed diagnoser can be converted to a simplified format, which is referred to in this paper as the *state-time tree*. This tree is essentially equivalent to the original version except that each node is identified with the newly reached equipment state(s) instead. For example, the markings in Figure 11 can first be replaced with place labels according to this principle. Every node in the state-time tree is then obtained by merging all nodes tagged with the same time stamp. By following this practice, Figure 11 can be transformed into Figure 15. Realistically speaking, only the nodes embedded in a state-time tree are distinguishable from one another. It should be noted that, although nodes [4], [5], and [7] are present at the same time, they are treated as distinct states. This is due to the assumption that transition t_{11} belongs to set $\mathbb{T}_K$, i.e., the corresponding controller signal is considered to be a piece of known information. Finally notice that nodes [10], [11], and [13] can be interpreted with the same approach.

Detectability, Diagnosability, and Resolution. From the special diagnoser structure constructed with the aforementioned procedure, it can be observed that the path associated with normal operation always contains a loop and a failed branch may either form a loop or end up at a single node. A failure can thus be detected when an observable system state is found to be abnormal (i.e., different from the normal operating conditions with the same time stamp) or permanently stays unchanged. Notice that the detectability of every fault origin in a diagnostic system, if badly designed, cannot always be guaranteed. For example, if the second and third activation conditions listed in Figure 4, i.e., LH and LL, are replaced with the *operation times* needed to fill up and empty the tank, respectively, then obviously none of the fault origins are detectable because the affected equipment states are not monitored.

In addition, since more than one scenario may produce the same online symptoms, the diagnosability of a *detectable* fault origin cannot be ensured either. A computer algorithm has thus been developed and coded with the programming language DELPHI to construct the aforementioned state-time

Table 2. Standard Format of Failure Signatures

branch no.	group no.	step (time)	fault origins	observable states	unobservable states	detectable	diagnosable
1	1	S_1 ($time_1$)	F_1	C1A, C1B, C1C, ...	c1a, c1b, c1c, ...	yes	yes
2	2	S_2 ($time_2$)	F_2	C2A, C2B, C2C, ...	c2a, c2b, c2c, ...	yes	no
3			F_3	C2A, C2B, C2C, ...	c3a, c3b, c3c, ...	yes	no
4	3	S_3 ($time_3$)	F_1, F_2	C4A, C4B, C4C, ...	c4a, c4b, c4c, ...	yes	yes
5	4	S_3 ($time_4$)	F_1, F_3	C5A, C5B, C5C, ...	c5a, c5b, c5c, ...	yes	yes

Table 3. Example 1 Data

(a) Normal Operation Sequence in Example 1

step (time)	observable states	unobservable states
$S_1(0)$	TL	V1C V2O OPNF IPF
$S_1(1)$	TH	V1C V2O OPNF IPF
$S_2(0)$	TH	V1O V2C OPF IPNF
$S_2(1)$	TL	V1O V2C OPF IPNF

(b) Failure Signatures in Example 1[a]

branch no.	group	step (time)	fault origins	node	observable states	unobservable states	diagnosable
1	1	$S_2(1)$	V1SC	24	TH	V1C V2C OPNF IPNF	yes
2	2	$S_1(1)$	V1SO	31	TL	V1O V2O OPF IPF	yes

[a] DTR: 100%. DGR: 100%. ANIB: 1.00.

tree and to search the resulting tree for a comprehensive diagnosability check (see the Supporting Information—part I). In a diagnosability check, the diagnosability of every detectable failed branch can be determined according to the recognizable events along the branch, i.e., the operation steps (times) and observable states. If this event sequence is unique, then the corresponding fault propagation scenario should be regarded as diagnosable. Otherwise, it is undiagnosable. Let us consider Figure 15 as an example to illustrate the classification criteria. Notice that, although the observable symptom of the first failed branch (nodes [6] and [8]) is the same as the normal conditions (nodes [4], [5], and [7]); the former is still detectable because nodes [6] and [8] have no child node, i.e., the system stays at the corresponding state indefinitely, while nodes [4], [5], and [7] are within a cyclic loop. Moreover, since this failed branch is unique, it is also diagnosable. Similarly, it can be argued the second failed branch (nodes [12] and [14]) should also be diagnosable.

As mentioned before, a detectable fault origin may not be diagnosable. It is thus necessary to use a quantitative measure of diagnostic resolution for comparing different candidate systems on a consistent basis. A logical choice for this purpose appears to be the average number of detectable faults sharing the identical online symptoms.

Performance Measures. A simple look-up table can be constructed to summarize the search results obtained with the diagnosability checks. The standard format of this table is shown in Table 2. Notice that the existence of failure(s) can be confirmed by comparing the *observable* system states along a failed branch (i.e., the failure signature) with the normal conditions. The fault origin(s) of every possible signature is listed in this table to facilitate efficient online fault identification. Thus, if the observable states of the abnormal system match those along branch 1 (identified at $time_1$ after issuing the controller command S_1), we could conclude that the failures F_1 have already occurred. On the other hand, F_2 and F_3 are indistinguishable after implementing S_2 since, although abnormal states can be detected at $time_2$ in both scenarios, the observable symptoms are identical. These two fault origins are thus placed in the same group.

As another example, let us consider the diagnoser in Figure 15. For comparison convenience, the normal operation sequence is presented in Table 3a and the corresponding failure signatures are shown in Table 3b. Notice that only the results obtained during *one* cyclic operation period are reported here, while the operation records in the initial period are omitted for the sake of conciseness. This is mainly due to the fact that the initial conditions are assigned arbitrarily

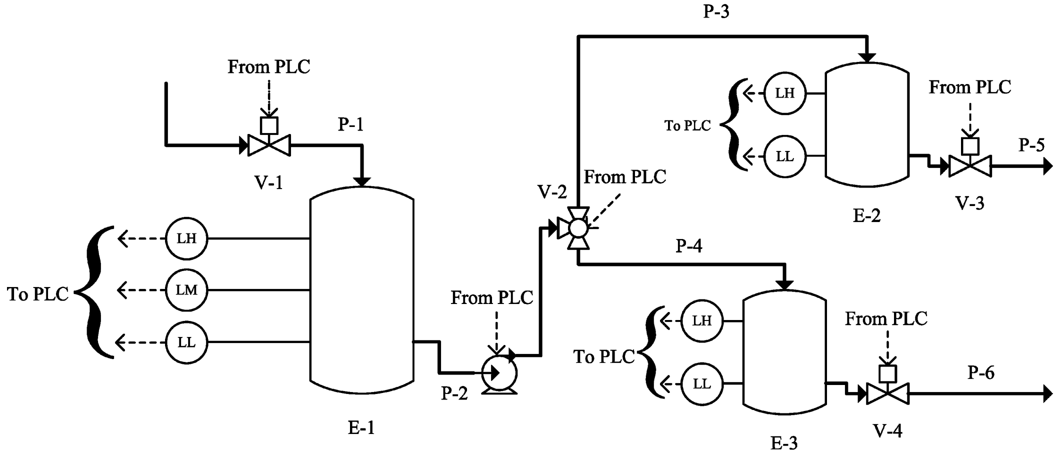

Figure 16. Three-tank system for illustrating the performance enhancement strategies (example 2).

Table 4. Operation Steps in Example 2

operation step	control actions
S_0	initialization
S_1	(1) open V-1; (2) close V-3; (3) close V-4
S_2	(1) close V-1; (2) switch V-2 to position +; (3) switch on E-4
S_3	switch V-2 to position −
S_4	(1) switch off E-4; (2) open V-3; (3) open V-4

Table 5. Activation Conditions of the Transitions in SFC of Example 2

transition	conditions
T_1	start
T_2	tank 1 LH
T_3	tank 2 LH
T_4	tank 3 LH
T_5	tank 1 LL, tank 2 LL, tank 3 LL

in this example. Consequently, the system states in the first period are different from those in the later periods, and therefore, the corresponding results are not representative. Notice also that the elapsed time of each step is given in the parentheses next to the step number in the first column of Table 3a and in the third column of Table 3b.

On the basis of the failure signatures along branches 1 and 2 in the diagnoser tree in Figure 15, all failures should be considered as diagnosable (see Table 3b). To further characterize the performance of a given diagnostic system, three quantitative measures have been devised in this study:

$$\text{detection rate(DTR)} = \left(1 - \frac{\text{number of undetectable branches}}{\text{total number of failed branches in reference case}}\right) \times 100\%$$

$$\text{diagnosis rate(DGR)} = \left(1 - \frac{\text{number of undiagnosable branches}}{\text{total number of failed branches in reference case}}\right) \times 100\%$$

$$\text{average number of indistinguishable branches(ANIB)} = \frac{\text{total number of failed branches in reference case}}{\text{number of indistinguishable groups}}$$

In these formulas, the so-called “reference case” refers to a batch system defined by the given P&ID and SFC (without any additional sensor or extra step which is not needed for normal operation). Notice that these measures can be computed with the results obtained from diagnosability check. For example, it can be determined by inspecting Table 3b that DTR, DGR, and ANIB equal 100%, 100%, and 1.00, respectively.

Performance Enhancement Strategies

The tasks of constructing a diagnoser for the system presented in Figure 1 and evaluating the corresponding diagnostic performance are trivial since only two failures of the same valve V-1 are considered as the possible fault origins. The issues concerning detectability, diagnosability, and resolution may

Table 6. Example 2 Data

(a) Normal Operation Sequence in Example 2

step (time)	observable states	unobservable states
$S_1(0)$	T1L T2L T3L	P1F P2NF P3NF P4NF P5NF P6NF P-OFF V1O V2- V3C V4C
$S_1(1)$	T1M T2L T3L	P1F P2NF P3NF P4NF P5NF P6NF P-OFF V1O V2- V3C V4C
$S_1(2)$	T1H T2L T3L	P1F P2NF P3NF P4NF P5NF P6NF P-OFF V1O V2- V3C V4C
$S_2(0)$	T1H T2L T3L	P1NF P2F P3F P4NF P5NF P6NF P-ON V1C V2+ V3C V4C
$S_2(1)$	T1M T2H T3L	P1NF P2NF P3NF P4NF P5NF P6NF P-ON V1C V2+ V3C V4C
$S_3(0)$	T1M T2H T3L	P1NF P2F P3NF P4F P5NF P6NF P-ON V1C V2- V3C V4C
$S_3(1)$	T1L T2H T3H	P1NF P2NF P3NF P4NF P5NF P6NF P-ON V1C V2- V3C V4C
$S_4(0)$	T1L T2H T3H	P1NF P2NF P3NF P4NF P5F P6F P-OFF V1C V2- V3O V4O
$S_4(1)$	T1L T2L T3L	P1NF P2NF P3NF P4NF P5NF P6NF P-OFF V1C V2- V3O V4O

(b) Failure Signatures in Example 2

branch no.	group	step (time)	fault origins	node	observable states	unobservable states	diagnosable
1	1	$S_1(1)$	V1SC	200	T1L T2L T3L	P1NF P2NF P3NF P4NF P5NF P6NF P-OFF V1C V2- V3C V4C	yes
2	2	$S_2(1)$	V1SO V3SO	251	T1H T2L T3L	P1F P2F P3F P4NF P5F P6NF P-ON V1O V2+ V3O V4C	yes
3	3	$S_2(1)$	V1SO	247	T1H T2H T3L	P1F P2NF P3NF P4NF P5NF P6NF P-ON V1O V2+ V3C V4C	yes
4	4	$S_2(1)$	V2S−	257	T1M T2L T3H	P1NF P2NF P3NF P4NF P5NF P6NF P-ON V1C V2- V3C V4C	no
5		$S_2(1)$	V2E−	258	T1M T2L T3H	P1NF P2NF P3NF P4NF P5NF P6NF P-ON V1C V2- V3C V4C	no
6	5	$S_2(1)$	T2LK	238	T1M T2L T3L	P1NF P2F P3F P4NF **P5NF** P6NF P-ON V1C V2+ V3C V4C	no
7		$S_2(1)$	V3SO	250	T1M T2L T3L	P1NF P2F P3F P4NF **P5F** P6NF P-ON V1C V2+ V3O V4C	no
8	6	$S_3(1)$	V2S+	269	T1M T2H T3L	P1NF P2NF P3NF P4NF P5NF P6NF P-ON V1C V2+ V3C V4C	no
9		$S_3(1)$	V2E+	263	T1M T2H T3L	P1NF P2NF P3NF P4NF P5NF P6NF P-ON V1C V2+ V3C V4C	no
10	7	$S_3(1)$	T3LK	177	T1L T2H T3L	P1NF P2NF P3NF P4NF P5NF P6NF P-ON V1C V2- V3C V4C	yes
11	8	$S_4(1)$	V3SC	193	T1L T2H T3L	P1NF P2NF P3NF P4NF P5NF P6NF P-OFF V1C V2- V3C V4O	yes

[a] DTR: 100%. DGR: 45.5%. ANIB: 1.38.

Table 7. Notations Used in Example 2

symbol	Description
T1H	tank 1 is at high level
T1M	tank 1 is at medium level
T1L	tank 1 is at low level
T2H	tank 2 is at high level
T2L	tank 2 is at low level
T2LK	leaks in tank 2
T3H	tank 3 is at high level
T3L	tank 3 is at low level
T3LK	leaks in tank 3
V1SO	valve 1 stuck at close position
V1SC	valve 1 stuck at open position
V2S+	valve 2 stuck at + position
V2S-	valve 2 stuck at − position
V2E+	valve 2 erroneously turn to + position
V2E-	valve 2 erroneously turn to − position
V3SO	valve 3 stuck at close position
V3SC	valve 3 stuck at open position
P1F	flow in pipeline 1
P1NF	no flow in pipeline 1
P2F	flow in pipeline 2
P2NF	no flow in pipeline 2
P3F	flow in pipeline 3
P3NF	no flow in pipeline 3
P4F	flow in pipeline 4
P4NF	no flow in pipeline 4
P5F	flow in pipeline 5
P5NF	no flow in pipeline 5
P6F	flow in pipeline 6
P6NF	no flow in pipeline 6

become critical if more complex system configurations and/or operating procedures are involved. Generally speaking, a sound diagnosis must be produced on the basis of sufficient online information. To this end, two fundamental approaches can be adopted to enhance the diagnostic performance of a given diagnoser at the design stage. One is to install additional sensors and the other is to implement additional operation steps which are not listed in the original SFC.

Second Illustrative Example. To illustrate the basic ideas of these two proposed strategies, let us consider the three-tank storage system in Figure 16. Pipeline P-1 is the inlet pipeline of tank E-1 and its flow is controlled with valve V-1. The outlet pipeline of E-1 is P-2, which is connected to a 3-way valve V-2, and pump E-4 is installed on P-2. When V-2 is at the position " + ", the fluid in P-2 will be transferred into pipeline P-3 and then tank E-2. If V-2 is at the position " − ", the fluid in P-2 will flow into pipeline P-4 and enter tank E-3. Valve V-3 is used to discharge the material in tank E-2, while V-4 is for E-3. Tank E-1 has a level sensor installed, and the sensor reports three conditions: (1) level low (LL); (2) level medium (LM); (3) level high (LH). The level sensors on tanks E-2 and E-3 can be used to detect only two conditions, i.e., (1) level low (LL) and (2) level high (LH). The detailed operation steps and their activation conditions are listed in Tables 4 and 5, respectively. Notice that these steps are supposed to be executed in sequence periodically.

In this example, the height of liquid level in every tank is assumed to be observable.The possible fault origins considered here are the following:

1. Valves V-1, V-2, and V-3 may experience sticking failures.
2. Valve V-2 may be switched to a wrong position due to spurious controller signal(s).
3. Tanks E-2 and E-3 may leak.

The Petri-net models of all components in this three-tank system (which includes 45 places, 57 transitions, and 234 arcs) can be found in the Supporting Information—part II. The time delays of all transitions in the tank models are assumed to be 1, while those in other component models are untimed.

The resulting normal operation sequence and the corresponding failure signatures are presented in Table 6a and b, respectively. The notations used in these tables are defined in Table 7. It can be observed from the signature list that, although some of the fault origins can be diagnosed correctly, many scenarios are still indistinguishable. Specifically, although all faults are detectable, the online symptoms of eleven possible scenarios can only be classified into eight distinct groups, and

Table 8. Failure Signatures in Example 2 with an Additional Flow Sensor on Pipelines 1 and 5[a]

branch no.	group	step (time)	fault origins	node	observable states	unobservable states	diagnosable
1	1	$S_1(0)$	V1SC	200	T1L T2L T3L P1NF P5NF	P2NF P3NF P4NF P6NF P-OFF V1C V2- V3C V4C	yes
2	2	$S_2(0)$	V1SO	231	T1H T2L T3L P1F P5NF	P2F P3F P4NF P6NF P-ON V1O V2+ V3C V4C	yes
3	3	$S_2(0)$	V3SO	242	T1H T2L T3L P1NF P5F	P2F P3F P4NF P6NF P-ON V1C V2+ V3O V4C	yes
4	4	$S_2(0)$	V1SO V3SO	251	T1H T2L T3L P1F P5F	P2F P3F P4NF P6NF P-ON V1O V2+ V3O V4C	yes
5	5	$S_2(1)$	T2LK	238	T1M T2L T3L P1NF P5NF	P2F P3F P4NF P6NF P-ON V1C V2+ V3C V4C	yes
6	6	$S_2(1)$	V2S−	257	T1M T2L T3H P1NF P5NF	P2NF P3NF P4NF P6NF P-ON V1C V2- V3C V4C	no
7			V2E−	258	T1M T2L T3H P1NF P5NF	P2NF P3NF P4NF P6NF P-ON V1C V2- V3C V4C	no
8	7	$S_3(1)$	V2S+	155	T1M T2H T3L P1NF P5NF	P2NF P3NF P4NF P6NF P-ON V1C V2+ V3C V4C	no
9			V2E+	149	T1M T2H T3L P1NF P5NF	P2NF P3NF P4NF P6NF P-ON V1C V2+ V3C V4C	no
10	8	$S_3(1)$	T3LK	177	T1L T2H T3L P1NF P5NF	P2NF P3NF P4NF P6NF P-ON V1C V2- V3C V4C	yes
11	9	$S_4(0)$	V3SC	181	T1L T2H T3H P1NF P5NF	P2NF P3NF P4NF P6F P-OFF V1C V2- V3C V4O	yes

[a] DTR: 100%. DGR: 63.6%. ANIB: 1.22.

Table 9. Diagnosis Results of Applying Extra Operation Steps for the Scenarios Listed in Table 6b[a]

branch no.	group	step (time)	fault origins	node	observable states	extra steps	new observable states	diagnosable
1	1	$S_1(1)$	V1SC	200	T1L T2L T3L			yes
2	2	$S_2(1)$	V1SO V3SO	251	T1H T2L T3L			yes
3	3	$S_2(1)$	V1SO	247	T1H T2H T3L			yes
4	4	$S_2(1)$	V2S−	257	T1M T2L T3H	V2+, wait	T1M T2L T3H	yes
5	5		V2E−	258	T1M T2L T3H	V2+, wait	**T1L T2H** T3H	yes
6	6	$S_2(1)$	T2LK	238	T1M T2L T3L	X		no
7			V3SO	250	T1M T2L T3L	X		no
8	7	$S_3(1)$	V2S+	269	T1M T2H T3L	V2−, wait	T1M T2H T3L	yes
9	8		V2E+	263	T1M T2H T3L	V2−, wait	**T1L** T2H **T3H**	yes
10	9	$S_3(1)$	T3LK	177	T1L T2H T3L			yes
11	10	$S_4(1)$	V3SC	193	T1L T2H T3L			yes

[a] DTR: 100%. DGR: 81.8%. ANIB: 1.10.

Table 10. Diagnosis Results of Applying Extra Operation Steps for the Scenarios Listed in Table 8[a]

branch no.	group	step (time)	fault origins	node	observable states	extra step	new observable states	diagnosable
1	1	$S_1(0)$	V1SC	200	T1L T2L T3L P1NF P5NF			yes
2	2	$S_2(0)$	V1SO	231	T1H T2L T3L P1F P5NF			yes
3	3	$S_2(0)$	V3SO	242	T1H T2L T3L P1NF P5F			yes
4	4	$S_2(0)$	V1SO V3SO	251	T1H T2L T3L P1F P5F			yes
5	5	$S_2(1)$	T2LK	238	T1M T2L T3L P1NF P5NF			yes
6	6	$S_2(1)$	V2S−	257	T1M T2L T3H P1NF P5NF	V2+, wait	T1M T2L T3H P1NF P5NF	yes
7	7		V2E−	258	T1M T2L T3H P1NF P5NF		**T1L T2H** T3H P1NF P5NF	yes
8	8	$S_3(1)$	V2S+	155	T1M T2H T3L P1NF P5NF	V2−, wait	T1M T2H T3L P1NF P5NF	yes
9	9		V2E+	149	T1M T2H T3L P1NF P5NF		**T1L** T2H **T3H** P1NF P5NF	yes
10	10	$S_3(1)$	T3LK	177	T1L T2H T3L P1NF P5NF			yes
11	11	$S_4(0)$	V3SC	181	T1L T2H T3H P1NF P5NF			yes

[a] DTR: 100%. DGR: 100%. ANIB: 1.00.

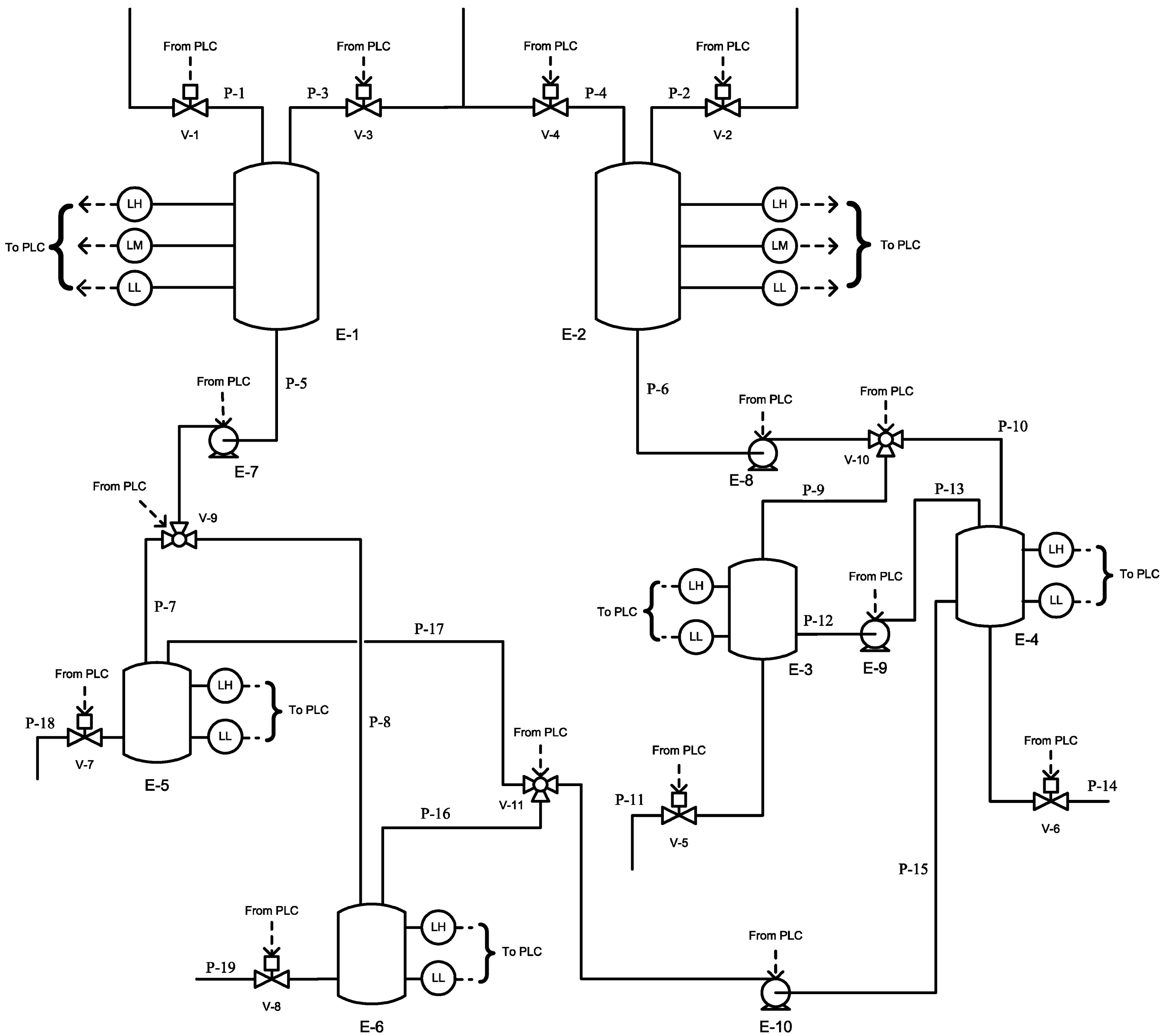

Figure 17. Six-tank storage system (example 3).

thus, those in groups 4, 5, and 6 cannot be diagnosed correctly. The corresponding detection rate (DTR), diagnosis rate (DGR), and average number of indistinguishable branches (ANIB) in this case are found to be 100%, 45.5%, and 1.38, respectively. This is clearly not satisfactory.

Selection of Extra Sensors. The most straightforward approach to enhance diagnostic performance is to install extra sensors. The obvious selection criterion is simply to measure the hidden state(s) which varies differently along different diagnoser branches in an indistinguishable group. For example, let us consider group 5 (which includes branches 6 and 7) in Table 6b. Notice that the unobservable states of pipeline P-5 are different in these two scenarios, i.e., P5NF and P5F (which are highlighted with boldface letters in the table). Thus, the corresponding fault origins could be diagnosed if a flow sensor is placed on P-5.

Since more than one indistinguishable group may appear in a diagnoser tree and, also, each group contains multiple origins, there is a need to determine the best locations of these extra sensors. This task is accomplished automatically with a brute-force approach, i.e., the diagnosability checks are performed for *all* possible combinations of the allowed sensor locations. This is primarily due to the fact that checking is a relatively simple task which can be completed almost instantaneously. Let us use the example mentioned above to illustrate the proposed approach. Let us assume that only flow sensors can be added and the maximum number of additional sensors is 2. By evaluating all possible options exhaustively, it was found that the best performance can be achieved by placing the flow sensors on pipeline 1 and pipeline 5. Table 8 is the updated signature list obtained after adding this sensor. Notice that the corresponding performance indices DGR and ANIB can be improved to 63.6% and 1.22, respectively.

Synthesis of Extra Operation Steps. Another approach to improve diagnostic performance is to carry out extra operation steps after detecting an abnormal system state. For example, let us consider group 6 in Table 6b, i.e., branches 8 and 9. Notice that all equipment states in these two scenarios are exactly the same, i.e., the corresponding fault origins cannot be separated from one another even with additional sensors. Thus, the only other way to enhance diagnostic resolution is to drive these identical states to some distinguishable conditions with extra operation steps. An exhaustive search strategy has been developed in this work to synthesize the needed diagnostic test procedure. To improve computation efficiency, a set of pruning rules have also been developed to reduce the search space dramatically. The detailed description of this algorithm is presented in the Appendix and the corresponding computer code can be found in the Supporting Information—part I.

By applying the proposed procedure, the extra steps for differentiating the aforementioned fault origins can be identified, that is, (1) switch valve V-2 to the − position and then (2) wait for a period of 1 time unit. It is obvious that the states on branch 8 cannot be changed with these actions since V-2 is stuck at the V2+ position. On the other hand, the position of valve V-2 can be altered in the case of branch 9 and, furthermore, the observable states should be converted from (T1M, T2H, T3L) to (T1L, T2H, T3H) after one unit time interval (see Table 9). It should also be pointed out that not every group of indistinguishable fault origins can be differentiated with a diagnostic test procedure. For example, let us consider group 5 in Table 6b. Notice that the observable direct outcomes of T2LK and V3SO are identical, i.e., the level of tank 2 is kept low in both cases, and there are no control actions that could be applied to differentiate these two scenarios. However, as mentioned before, this problem can of course be solved by adding an additional flow sensor on pipeline 5.

The search results of all the cases listed in Tables 6b and 8 are summarized in Tables 9 and 10, respectively. Notice that the boldfaced symbols denote the distinguishable states created with extra steps. From these results, it can be found that the DGR can be dramatically increased to 81.8% and 100%, respectively, and ANIB values become 1.10 and 1.00.

Applications

A more complex problem (see Figure 17) is considered here to demonstrate the feasibility and effectiveness of the proposed approach to configure appropriate fault diagnosis systems for large batch processes. A brief process description is first provided in the following.

Table 11. Operation Steps in Example 3

operation step	control actions
S_0	initialization
S_1	(1) open V-1; (2) open V-2; (3) close V-7; (4) close V-8
S_2	(1) close V-1; (2) close V-2; (3) switch V-10 to position −; (4) switch on E-8
S_3	(1) switch off E-8; (2) switch V-9 to position +; (3) switch on E-7
S_4	(1) switch off E-7; (2) switch V-10 to position +; (3) switch on E-8
S_5	(1) switch off E-8; (2) switch V-11 to position −; (3) switch on E-10
S_6	(1) switch off E-10; (2) open V-7; (3) switch on E-9
S_7	(1) switch off E-9; (2) close V-7; (3) switch V-11 to position +; (4) switch on E-10
S_8	(1) switch off E-10; (2) open V-7; (3) open V-8

Table 12. Activation Conditions of the Transitions in SFC of Example 3

transition	conditions
T_1	start
T_2	tank 1 LH, tank 2 LH
T_3	tank 4 LH, tank 2 LM
T_4	tank 5 LH, tank 1 LM
T_5	tank 3 LH, tank 2 LL
T_6	tank 6 LH, tank 4 LL
T_7	tank 3 LL, tank 4 LH, tank 5 LL
T_8	tank 5 LH, tank 4 LL
T_9	tank 5 LL, tank 6 LL

Process Description. A six-tank storage system is considered here. P-1 and P-3 are the inlet pipelines of tank E-1, and their flows are manipulated with valves V-1 and V-3, respectively. The outlet pipeline of E-1 is P-5, on which is pump E-7 is installed. Pipeline P-5 ends at the three-way valve V-9. When V-9 is at the position +, the fluid in P-5 will be transferred into pipeline P-7 and then tank E-5. If V-9 is at position −, the fluid in P-5 will transferred into pipeline P-8 and enter tank E-6. Valve V-7 is used to discharge the material in tank E-5, while V-8 is for E-6.

On the other hand, pipelines P-2 and P-4 are the inlet pipelines of tank E-2 and their flows are controlled with valves V-2 and V-4, respectively. The outlet pipeline of E-2 is P-6, on which pump E-8 is installed. Pipeline P-6 ends at the 3-way valve V-10. When V-10 is at position +, the fluid in P-6 will be transferred into pipeline P-9 and then tank E-3. If V-10 is at position −, the fluid in P-6 will flow into pipeline P-9 and enter tank E-4. Valve V-5 is used to discharge the material in tank E-3, while V-6 is for E-4.

The fluid in E-3 can be transferred to E-4 with pump E-9 via pipelines P-12 and P-13, while the fluid in E-4 can be

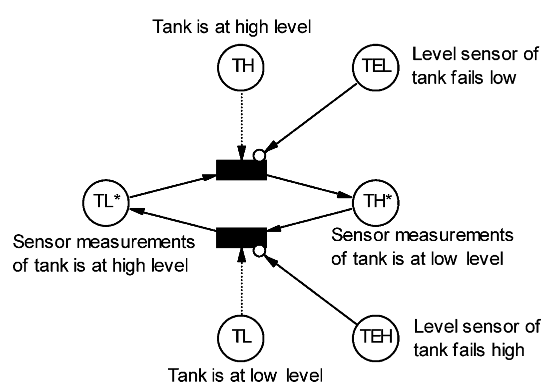

Figure 18. Petri-net model of level sensor.

Table 13. Example 3 Data

(a) Normal Operation Sequence in Example 3

step (time)	event states	observable states
$S_1(0)$	P1F P2F V1O V2O V7C V8C	T1M T2L T3L* T4L* T5L* T6L*
$S_1(1)$	T1H T2M	T1H T2M T3L* T4L* T5L* T6L*
$S_1(2)$	T2H	T1H T2H T3L* T4L* T5L* T6L*
$S_2(0)$	P1NF P2NF P6F P10F V1C V2C V10− PP2-ON	T1H T2H T3L* T4L* T5L* T6L*
$S_2(1)$	T2M T4H P6NF P10NF T4H*	T1H T2M T3L* T4H* T5L* T6L*
$S_3(0)$	P5F P7F PP1-ON PP2-OFF	T1H T2M T3L* T4H* T5L* T6L*
$S_3(1)$	T1M T5H P5NF P7NF T5H*	T1M T2M T3L* T4H* T5H* T6L*
$S_4(0)$	P6F P9F V10+ PP1-OFF PP2-ON	T1M T2M T3L* T4H* T5H* T6L*
$S_4(1)$	T2L T3H P6NF P9NF T3H*	T1M T2L T3H* T4H* T5H* T6L*
$S_5(0)$	P15F P16F V11− PP2-OFF PP4-ON	T1M T2L T3H* T4H* T5H* T6L*
$S_5(1)$	T4L T6H P15NF P16NF T4L* T6H*	T1M T2L T3H* T4L* T5H* T6H*
$S_6(0)$	P12F P13F P18F V7O PP3-ON PP4-OFF	T1M T2L T3H* T4L* T5H* T6H*
$S_6(1)$	T3L T4H T5L P12NF P13NF P18NF T3L* T4H* T5L*	T1M T2L T3L* T4H* T5L* T6H*
$S_7(0)$	P15F P17F V7C V11+ PP3-OFF PP4-ON	T1M T2L T3L* T4H* T5L* T6H*
$S_7(1)$	T4L T5H P15NF P17NF T4L* T5H*	T1M T2L T3L* T4L* T5H* T6H*
$S_8(0)$	P18F P19F V7O V8O PP4-OFF	T1M T2L T3L* T4L* T5H* T6H*
$S_8(1)$	T5L T6L P18NF P19NF T5L* T6L*	T1M T2L T3L* T4L* T5L* T6L*

(b) Failure Signatures in Example 3[a]

branch no.	group	step (time)	fault origins	node	observable states	unobservable states	diagnosable
1	1	$S_1(1)$	T2LK	485	T1H T2L T3L* T4L* T5L* T6L*	T3L T4L T5L T6L P1F P2F P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1O V2O V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-OFF	no
2			V2SC	487	T1H T2L T3L* T4L* T5L* T6L*	T3L T4L T5L T6L P1F P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1O V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-OFF	no
3	2	$S_1(1)$	V1SC	481	T1M T2M T3L* T4L* T5L* T6L*	T3L T4L T5L T6L P1NF P2F P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2O V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-OFF	yes
4	3	$S_2(1)$	T4EL	559	T1H T2M T3L* T4L* T5L* T6L*	T3L T4H T5L T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-OFF PP2-ON PP3-OFF PP4-OFF	no
5			T4LK	521	T1H T2M T3L* T4L* T5L* T6L*	T3L T4L T5L T6L P1NF P2NF P3NF P4NF P5NF P6F P7NF P8NF P9NF P10F P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-OFF PP2-ON PP3-OFF PP4-OFF	no
6	4	$S_2(1)$	V2SO	562	T1H T2H T3L* T4H* T5L* T6L*	T3L T4H T5L T6L P1NF P2F P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2O V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-OFF PP2-ON PP3-OFF PP4-OFF	yes
7	5	$S_2(1)$	V10S+	563	T1H T2M T3H* T4L* T5L* T6L*	T3H T4L T5L T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-ON PP3-OFF PP4-OFF	no
8			V10E+	564	T1H T2M T3H* T4L* T5L* T6L*	T3H T4L T5L T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-ON PP3-OFF PP4-OFF	no
9	6	$S_3(1)$	V1SO T5LK	606	T1H T2M T3L* T4H* T5L* T6L*	T3L T4H T5L T6L P1F P2NF P3NF P4NF P5F P6NF P7F P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1O V2C V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-ON PP2-OFF PP3-OFF PP4-OFF	no
10			V1SO V7SO	609	T1H T2M T3L* T4H* T5L* T6L*	T3L T4H T5L T6L P1F P2NF P3NF P4NF P5F P6NF P7F P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18F P19NF V1O V2C V3C V4C V5C V6C V7O V8C V9+ V10− V11+ PP1-ON PP2-OFF PP3-OFF PP4-OFF	no
11	7	$S_3(1)$	T5EL	612	T1M T2M T3L* T4H* T5L* T6L*	T3L T4H T5H T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-ON PP2-OFF PP3-OFF PP4-OFF	no

Table 13 Continued

(b) Failure Signatures in Example 3[a]

branch no.	group	step (time)	fault origins	node	observable states	unobservable states	diagnosable
12			T5LK	593	T1M T2M T3L* T4H* T5L* T6L*	T3L T4H T5L T6L P1NF P2NF P3NF P4NF P5F P6NF P7F P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-ON PP2-OFF PP3-OFF PP4-OFF	no
13			V7SO	607	T1M T2M T3L* T4H* T5L* T6L*	T3L T4H T5L T6L P1NF P2NF P3NF P4NF P5F P6NF P7F P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18F P19NF V1C V2C V3C V4C V5C V6C V7O V8C V9+ V10− V11+ PP1-ON PP2-OFF PP3-OFF PP4-OFF	no
14	8	$S_3(1)$	V9E−	614	T1M T2M T3L* T4H* T5L* T6H*	T3L T4H T5L T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9- V10− V11+ PP1-ON PP2-OFF PP3-OFF PP4-OFF	yes
15	9	$S_3(1)$	V1SO	615	T1H T2M T3L* T4H* T5H* T6L*	T3L T4H T5H T6L P1F P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1O V2C V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-ON PP2-OFF PP3-OFF PP4-OFF	yes
16	10	$S_3(1)$	V1SO T5EL	616	T1H T2M T3L* T4H* T5L* T6L*	T3L T4H T5H T6L P1F P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1O V2C V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-ON PP2-OFF PP3-OFF PP4-OFF	yes
17	11	$S_4(1)$	V10S−	625	T1M T2M T3L* T4H* T5H* T6L*	T3L T4H T5H T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-OFF PP2-ON PP3-OFF PP4-OFF	no
18			V10E−	626	T1M T2M T3L* T4H* T5H* T6L*	T3L T4H T5H T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10− V11+ PP1-OFF PP2-ON PP3-OFF PP4-OFF	no
19	12	$S_4(1)$	T3EL	380	T1M T2L T3L* T4H* T5H* T6L*	T3H T4H T5H T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-ON PP3-OFF PP4-OFF	no
20			T3LK	384	T1M T2L T3L* T4H* T5H* T6L*	T3L T4H T5H T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-ON PP3-OFF PP4-OFF	no
21	13	$S_5(1)$	V11S+	386	T1M T2L T3H* T4H* T5H* T6L*	T3H T4H T5H T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-ON	yes
22	14	$S_5(1)$	T4EH	399	T1M T2L T3H* T4H* T5H* T6H*	T3H T4L T5H T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11− PP1-OFF PP2-OFF PP3-OFF PP4-ON	yes
23	15	$S_5(1)$	T6EL	400	T1M T2L T3H* T4L* T5H* T6L*	T3H T4L T5H T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11− PP1-OFF PP2-OFF PP3-OFF PP4-ON	no
24			T6LK	401	T1M T2L T3H* T4L* T5H* T6L*	T3H T4L T5H T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11− PP1-OFF PP2-OFF PP3-OFF PP4-ON	no
25	16	$S_6(1)$	T3EH	424	T1M T2L T3H* T4H* T5L* T6H*	T3L T4H T5L T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7O V8C V9+ V10+ V11− PP1-OFF PP2-OFF PP3-ON PP4-OFF	yes
26	17	$S_6(1)$	T4EL	425	T1M T2L T3L* T4L* T5L* T6H*	T3L T4H T5L T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7O V8C V9+ V10+ V11− PP1-OFF PP2-OFF PP3-ON PP4-OFF	no

Table 13 Continued

(b) Failure Signatures in Example 3[a]

branch no.	group	step (time)	fault origins	node	observable states	unobservable states	diagnosable
27			T4LK	427	T1M T2L T3L* T4L* T5L* T6H*	T3L T4L T5L T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7O V8C V9+ V10+ V11− PP1-OFF PP2-OFF PP3-ON PP4-OFF	no
28	18	$S_6(1)$	T5EH	426	T1M T2L T3L* T4H* T5H* T6H*	T3L T4H T5L T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7O V8C V9+ V10+ V11− PP1-OFF PP2-OFF PP3-ON PP4-OFF	no
29			V7SC	428	T1M T2L T3L* T4H* T5H* T6H*	T3L T4H T5H T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11− PP1-OFF PP2-OFF PP3-ON PP4-OFF	no
30	19	$S_7(1)$	V11S-	434	T1M T2L T3L* T4H* T5L* T6H*	T3L T4H T5L T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11− PP1-OFF PP2-OFF PP3-OFF PP4-ON	yes
31	20	$S_7(1)$	T4EH	451	T1M T2L T3L* T4H* T5H* T6H*	T3L T4L T5H T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-ON	yes
32	21	$S_7(1)$	T5EL	452	T1M T2L T3L* T4L* T5L* T6H*	T3L T4L T5H T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-ON	no
33			T5LK	453	T1M T2L T3L* T4L* T5L* T6H*	T3L T4L T5L T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8C V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-ON	no
34			V7SO	460	T1M T2L T3L* T4L* T5L* T6H*	T3L T4L T5L T6H P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7O V8C V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-ON	no
35	22	$S_8(1)$	T5EH	468	T1M T2L T3L* T4L* T5H* T6L*	T3L T4L T5L T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7O V8O V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-OFF	no
36			V7SC	470	T1M T2L T3L* T4L* T5H* T6L*	T3L T4L T5H T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7C V8O V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-OFF	no
37	23	$S_8(1)$	T6EH	469	T1M T2L T3L* T4L* T5L* T6H*	T3L T4L T5L T6L P1NF P2NF P3NF P4NF P5NF P6NF P7NF P8NF P9NF P10NF P11NF P12NF P13NF P14NF P15NF P16NF P17NF P18NF P19NF V1C V2C V3C V4C V5C V6C V7O V8O V9+ V10+ V11+ PP1-OFF PP2-OFF PP3-OFF PP4-OFF	yes

[a] DTR: 100%. DGR: 29.7%. ANIB: 1.61.

transported to tank E-5 and E-6 with pump E-10 via pipeline P-15. Notice that P-15 ends at the 3-way valve V-11. When V-11 is at position +, the fluid in P-11 will be transferred into pipeline P-17 and then tank E-5. If V-11 is at position −, the fluid in P-11 will flow into pipeline P-16 and enter tank E-6.

Both E-1 and E-2 are equipped with level sensors. Each reposts three conditions: (1) level low (LL); (2) level medium (LM); (3) level high (LH). On the other hand, the level sensors on tanks E-3, E-4, E-5, and E-6 can be used to detect only two conditions, i.e., (1) level low (LL) and (2) level high (LH). The fault origins considered in this example are the following:

1. Valves V-1, V-2, V-5, V-6, V-7, V-9, V-10, and V-11 may experience sticking failures.
2. Valve V-9 and V-10 may be switched to a wrong position due to spurious controller signal(s).
3. Tanks E-1, E-2, E-3, E-4, E-5, and E-6 may leak.
4. The measurements of level sensors on E-3, E-4, E-5, and E-6 may be erroneous.

The detailed operation steps in this procedure and their activation conditions are listed in Tables 11 and 12, respectively. Notice that these steps are also executed in sequence periodically.

Petri-net Models. Since almost all component models adopted in the present example are essentially the same as those used previously, descriptions of these models are not repeated here for the sake of brevity. On the other hand, notice that the sensor malfunctions have not been considered before. It is thus

Table 14. Notations Used in Example 3

symbol	description	symbol	description
T1H	tank 1 is at high level	V9S−	valve 9 stuck at − position
T1M	tank 1 is at medium level	V9E+	valve 9 erroneously turned to + position
T1L	tank 1 is at low level	V9E−	valve 9 erroneously turned to − position
T2H	tank 2 is at high level	V10S+	valve 10 stuck at + position
T2L	tank 2 is at low level	V10S−	valve 10 stuck at − position
T2LK	leaks in tank 2	V10E+	valve 10 erroneously turned to + position
T3H	tank 3 is at high level	V10E−	valve 10 erroneously turned to − position
T3L	tank 3 is at low level	V11S+	valve 11 stuck at + position
T3H*	sensor measurements of tank 3 is at high level	V11S−	valve 11 stuck at − position
T3L*	sensor measurements of tank 3 is at low level	P1F	flow in pipeline 1
T3EH	level sensor of tank 3 fails high	P1NF	no flow in pipeline 1
T3EL	level sensor of tank 3 fails low	P2F	flow in pipeline 2
T3LK	leaks in tank 3	P2NF	no flow in pipeline 2
T4H	tank 4 is at high level	P3F	flow in pipeline 3
T4L	tank 4 is at low level	P3NF	no flow in pipeline 3
T4H*	sensor measurements of tank 4 is at high level	P4F	flow in pipeline 4
T4L*	sensor measurements of tank 4 is at low level	P4NF	no flow in pipeline 4
T4EH	level sensor of tank 4 fails high	P5F	flow in pipeline 5
T4EL	level sensor of tank 4 fails low	P5NF	no flow in pipeline 5
T4LK	leaks in tank 4	P6F	flow in pipeline 6
T5H	tank 5 is at high level	P6NF	no flow in pipeline 6
T5L	tank 5 is at low level	P7F	flow in pipeline 7
T5H*	sensor measurements of tank 5 is at high level	P7NF	no flow in pipeline 7
T5L*	sensor measurements of tank 5 is at low level	P8F	flow in pipeline 8
T5EH	level sensor of tank 5 fails high	P8NF	no flow in pipeline 8
T5EL	level sensor of tank 5 fails low	P9F	flow in pipeline 9
T5LK	leaks in tank 5	P9NF	no flow in pipeline 9
T6H	tank 6 is at high level	P10F	flow in pipeline 10
T6L	tank 6 is at low level	P10NF	no flow in pipeline 10
T6H*	sensor measurements of tank 6 is at high level	P11F	flow in pipeline 11
T6L*	sensor measurements of tank 6 is at low level	P11NF	no flow in pipeline 11
T6EH	level sensor of tank 6 fails high	P12F	flow in pipeline 12
T6EL	level sensor of tank 6 fails low	P12NF	no flow in pipeline 12
T6LK	leaks in tank 6	P13F	flow in pipeline 13
V1SO	valve 1 stuck at close position	P13NF	no flow in pipeline 13
V1SC	valve 1 stuck at open position	P14F	flow in pipeline 14
V2SO	valve 2 stuck at close position	P14NF	no flow in pipeline 14
V2SC	valve 2 stuck at open position	P15F	flow in pipeline 15
V5SO	valve 5 stuck at close position	P15NF	no flow in pipeline 15
V5SC	valve 5 stuck at open position	P16F	flow in pipeline 16
V6SO	valve 6 stuck at close position	P16NF	no flow in pipeline 16
V6SC	valve 6 stuck at open position	P17F	flow in pipeline 17
V7SO	valve 7 stuck at close position	P17NF	no flow in pipeline 17
V7SC	valve 7 stuck at open position	P18F	flow in pipeline 18
V9S+	valve 9 stuck at + position	P18NF	no flow in pipeline 18

Table 15. Values of DGR and ANIB Achieved with Different Combinations of Additional Sensors in Example 3

number of sensors	DGR	ANIB	combinations of additional sensors
0	29.7	1.61	
1	45.9	1.37	[P18]
2	51.4	1.32	[P2, P18] [P5, P18] [P6, P18] [P7, P18] [P10, P18]
3	56.8	1.28	[P2, P5, P18] [P2, P6, P18] [P2, P7, P18] [P2, P10, P18] [P5, P6, P18] [P5, P10, P18] [P6, P7, P18] [P7, P10, P18]
4	62.2	1.23	[P2, P5, P6, P18] [P2, P5, P10, P18] [P2, P6, P7, P18] [P2, P7, P10, P18]

necessary to develop an additional Petri-net for representing the level sensors on tanks E-3, E-4, E-5, and E-6 (see Figure 18). In this model, places TH and TL denote the actual liquid levels, TH* and TL* represent the corresponding sensor measurements, while TEH and TEL are used to model sensor failures which result in constantly high and low output respectively. Finally, it should be noted that the resulting system model contains 137 places, 176 transitions, and 708 arcs.

Diagnoser Performance. By constructing the corresponding diagnoser tree, the normal operation sequence and the corresponding failure signatures can be identified (see Table 13a and b). Notice also that all notations used in this example are defined in Table 14. Since DGR = 29.7% and ANIB = 1.61, the corresponding diagnostic performance does not seem to be satisfactory.

Diagnosability checks have also been carried out to determine the performance indices, i.e., DGR and ANIB, achieved with different combinations of additional sensors. The corresponding values of DGR and ANIB can be found in Table 15. Notice that installing a flow sensor on pipeline 18 would enhance DGR and ANIB to 45.9% and 1.37, respectively. This addition could obviously produce the most significant improvement in diagnostic performance with the lowest expenditure, while adding any more sensors could only improve the system performance marginally.

As indicated before, the next task is to synthesize and then implement the diagnostic test procedure. The search results of the cases listed in Table 13b are summarized in Table 16. From these results, it can be found that the DGR and ANIB are dramatically raised to 83.8% and 1.06. Moreover, if the proposed search procedure is applied to the same system with an extra flow sensor on pipeline 18, the DGR can be improved to 100%.

Table 16. Diagnosis Results of Applying Extra Operation Steps in Table 13b in Example 3[a]

branch no.	group	step (time)	fault origins	node	observable states	extra steps	new observable state	diagnosable
1	1	$S_1(1)$	T2LK	485	T1H T2L T3L* T4L* T5L* T6L*	V4O, wait	T1H T2L T3L* T4L* T5L* T6L*	yes
2	2		V2SC	487	T1H T2L T3L* T4L* T5L* T6L*	V4O, wait	T1H **T2M** T3L* T4L* T5L* T6L*	yes
3	3	$S_1(1)$	V1SC	481	T1M T2M T3L* T4L* T5L* T6L*			yes
4	4	$S_2(1)$	T4EL	559	T1H T2M T3L* T4L* T5L* T6L*	wait	T1H T2M T3L* T4L* T5L* T6L*	yes
5	5		T4LK	521	T1H T2M T3L* T4L* T5L* T6L*	wait	T1H **T2L** T3 L* T4L* T5L* T6L*	yes
6	6	$S_2(1)$	V2SO	562	T1H T2H T3L* T4H* T5L* T6L*			yes
7	7	$S_2(1)$	V10S+	563	T1H T2M T3H* T4L* T5L* T6L*	V10−, wait	T1H T2M T3H* T4L* T5L* T6L*	yes
8	8		V10E+	564	T1H T2M T3H* T4L* T5L* T6L*	V10−, wait	T1H **T2L** T3H* **T4H*** T5L* T6L*	yes
9	9	$S_3(1)$	V1SO T5LK	606	T1H T2M T3L* T4H* T5L* T6L*	X		no
10	10		V1SO V7SO	609	T1H T2M T3L* T4H* T5L* T6L*	X		no
11	11	$S_3(1)$	T5EL	612	T1M T2M T3L* T4H* T5L* T6L*	wait	T1M T2M T3L* T4H* T5L* T6L*	yes
12	12		T5LK	593	T1M T2M T3L* T4H* T5L* T6L*	wait	**T1L** T2M T3L* T4H* T5L* T6L*	no
13			V7SO	607	T1M T2M T3L* T4H* T5L* T6L*	wait	**T1L** T2M T3L* T4H* T5L* T6L*	no
14	13	$S_3(1)$	V9E−	614	T1M T2M T3L* T4H* T5L* T6H*			yes
15	14	$S_3(1)$	V1SO	615	T1H T2M T3L* T4H* T5H* T6L*			yes
16	15	$S_3(1)$	V1SO T5EL	616	T1H T2M T3L* T4H* T5L* T6L*			yes
17	16	$S_4(1)$	V10S−	625	T1M T2M T3L* T4H* T5H* T6L*	V10+, wait	T1M T2M T3L* T4H* T5H* T6L*	yes
18	17		V10E−	626	T1M T2M T3L* T4H* T5H* T6L*	V10+, wait	T1M **T2L** **T3H*** T4H* T5H* T6L*	yes
19	18	$S_4(1)$	T3EL	380	T1M T2L T3L* T4H* T5H* T6L*	V2O, wait	T1M **T2M** T3L* T4H* T5H* T6L*	yes
20	19		T3LK	384	T1M T2L T3L* T4H* T5H* T6L*	V2O, wait	T1M T2L T3L* T4H* T5H* T6L*	yes
21	20	$S_5(1)$	V11S+	386	T1M T2L T3H* T4H* T5H* T6L*			yes
22	21	$S_5(1)$	T4EH	399	T1M T2L T3H* T4H* T5H* T6H*			yes
23	22	$S_5(1)$	T6EL	400	T1M T2L T3H* T4L* T5H* T6L*	V9−, PP1-ON, wait	T1M T2L T3H* T4L* T5H* T6L*	yes
24	23		T6LK	401	T1M T2L T3H* T4L* T5H* T6L*	V9−, PP1-ON, wait	**T1L** T2L T3H* T4L* T5H* T6L*	yes
25	24	$S_6(1)$	T3EH	424	T1M T2L T3H* T4H* T5L* T6H*			yes
26	25	$S_6(1)$	T4EL	425	T1M T2L T3L* T4L* T5L* T6H*	V3O, V9−, PP1-ON, wait	T1M T2L T3L* T4L* T5L* T6H*	yes
27	26		T4LK	427	T1M T2L T3L* T4L* T5L* T6H*	V3O, V9−, PP1-ON, wait	**T1H** T2L T3L* T4L* T5L* T6H*	yes
28	27	$S_6(1)$	T5EH	426	T1M T2L T3L* T4H* T5H* T6H*	PP1-ON, wait	**T1L** T2L T3L* T4H* T5H* T6H*	yes
29	28		V7SC	428	T1M T2L T3L* T4H* T5H* T6H*	PP1-ON, wait	T1M T2L T3L* T4H* T5H* T6H*	yes
30	29	$S_7(1)$	V11S−	434	T1M T2L T3L* T4H* T5L* T6H*			yes
31	30	$S_7(1)$	T4EH	451	T1M T2L T3L* T4H* T5H* T6H*			yes
32	31	$S_7(1)$	T5EL	452	T1M T2L T3L* T4L* T5L* T6H*	PP1-ON, wait	T1M T2L T3L* T4L* T5L* T6H*	yes
33	32		T5LK	453	T1M T2L T3L* T4L* T5L* T6H*	PP1-ON, wait	**T1L** T2L T3L* T4L* T5L* T6H*	no
34			V7SO	460	T1M T2L T3L* T4L* T5L* T6H*	PP1-ON, wait	**T1L** T2L T3L* T4L* T5L* T6H*	
35	33	$S_8(1)$	T5EH	468	T1M T2L T3L* T4L* T5H* T6L*	PP1-ON, wait	**T1L** T2L T3L* T4L* T5H* T6L*	yes
36	34		V7SC	470	T1M T2L T3L* T4L* T5H* T6L*	PP1-ON, wait	T1M T2L T3L* T4L* T5H* T6L*	yes
37	35	$S_8(1)$	T6EH	469	T1M T2L T3L* T4L* T5L* T6H*			yes

[a] DTR: 100%. DGR: 83.8%. ANIB: 1.06.

This is due to the fact that there are only three indistinguishable groups after performing the diagnostic tests suggested for the original system (see Table 16), but all of them can be differentiated by adding a flow sensor on pipeline P-18.

Conclusions

A systematic Petri-net based procedure is presented in this paper to design the online fault diagnosis systems for batch processes. Specifically, this procedure consists of the following tasks:

- Build a Petri-net model based on the given process flow diagram (P&ID) and operating procedure (SFC).
- Construct the corresponding diagnoser tree to evaluate the detectability, diagnosability, and resolution of the given system.
- If necessary, implement the proposed performance enhancement strategies to select additional sensors and synthesize the diagnostic test procedures.

It should be noted that computer programs have already been developed in this work to perform the last two tasks automatically. Thus, configuration of fault diagnosis systems for realistic batch processes becomes efficient and error-free.

Appendix: Exhaustive Search Algorithm for Generating Diagnostic Test Procedure

As mentioned previously, the extra operation steps are generated in this work with an enhanced exhaustive search strategy. If the number of possible actions at a given system state is X and the maximum number of extra *one-action* steps allowed in the diagnostic test procedure is N, then there should be X^N different combinations to be evaluated. Since the efficiency of this algorithm is clearly very poor, the following pruning rules have been adopted to reduce the search space, i.e.

1. Examine only enabled actions.
2. Neglect actions having no net effects.
3. Ignore symmetrical actions.

For illustration convenience, let us consider the system presented in Figure 1 and two indistinguishable scenarios: (1) valve V-1 is closed mistakenly (V1EC) and valve V-2 is stuck at open position (V2SO), and (2) valve V-1 is closed mistakenly (V1EC) and valve V-2 is opened mistakenly (V2EO). In both cases, the liquid level should stay at the high position after executing the actions in S_2 and all other unobservable states are identical.

There are five possible actions in the system under consideration: (1) open V-1 (V1O), (2) close V-1 (V1C), (3) open V-2 (V2O), (4) close V-2 (V2C), and (5) wait for 1 time unit (wait). The first step in the diagnostic test procedure can be selected according to Figure A.1. Among the five possible actions, V1C, V2O, and wait can be excluded according to rule 1. Consequently, the first level of the pruned search tree becomes the one shown in Figure A.2. However, after implementing the remaining two actions individually, it can be found that the aforementioned fault origins are still indistinguishable. It is therefore necessary to implement more than one step in the diagnostic test procedure.

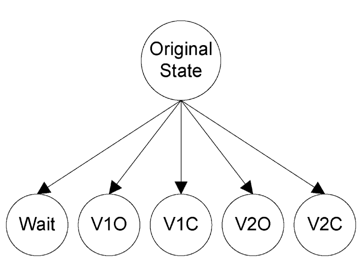

Figure A.1. One-level exhaustive search tree.

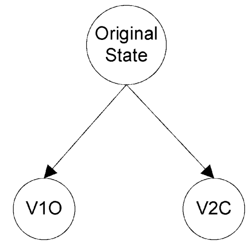

Figure A.2. One-level search tree obtained with rule 1.

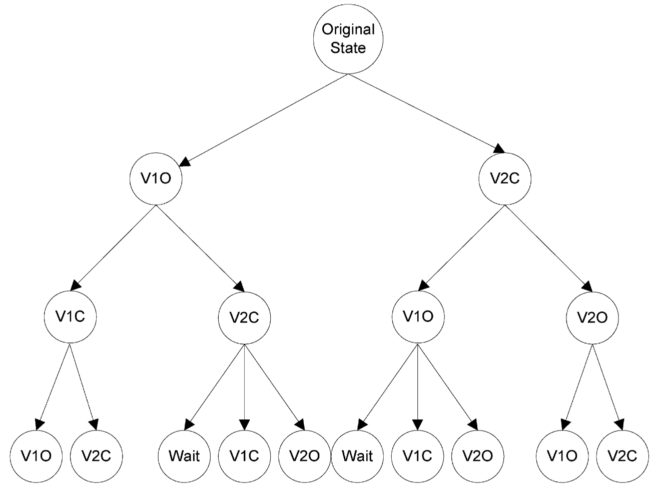

Figure A.3. Three-level search tree obtained with rule 1.

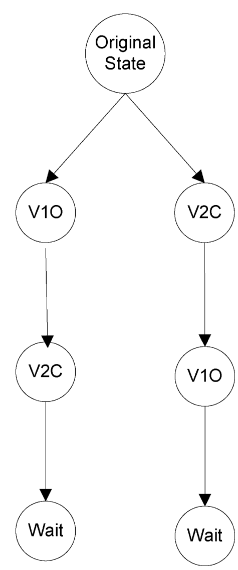

Figure A.4. Three-level search tree obtained with rules 1 and 2.

Let us try to synthesize a procedure with up to 3 steps, i.e., $N = 3$. Specifically, by following rule 1 only, the three-level search tree in Figure A.3 can be constructed from the tree in Figure A.2. By applying rule 2, the paths V1O−V1C and V2C−V2O in Figure A.3 can be deleted. Moreover, it is obvious that the system state reached by following the branch V1O−V2C−V1C must be the same as that at V2C and, thus, the search tree can be further pruned to the one shown in Figure A.4. Notice finally that the implementation order of actions V1O and V2C is irrelevant if they are carried out consecutively. The same system state can be reached instantaneously in both cases

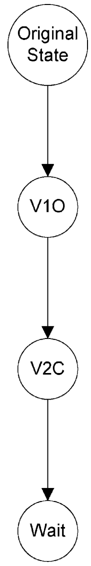

Figure A.5. Three-level search tree obtained with rules 1−3.

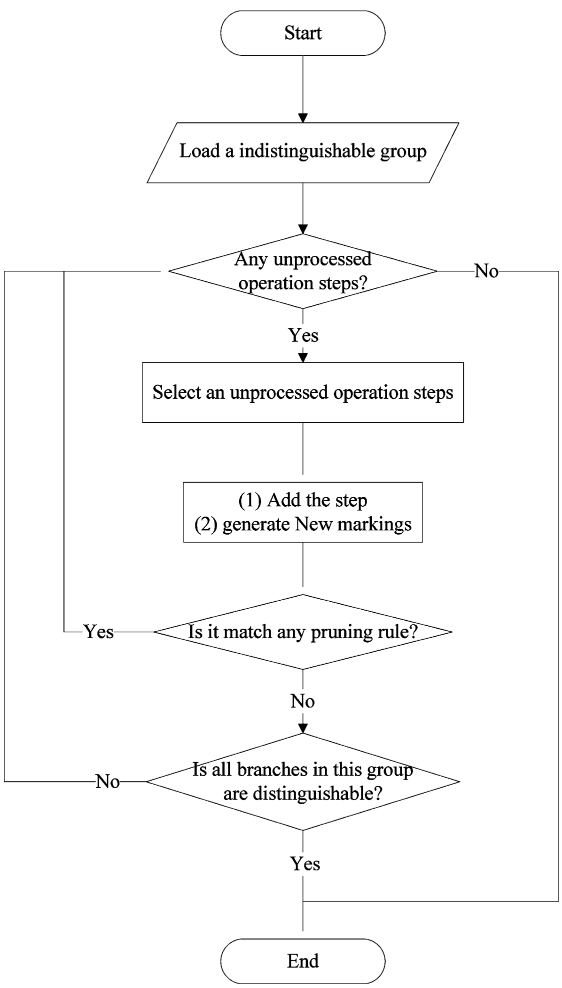

Figure A.6. Flowchart of the synthesis algorithm for the diagnostic test procedure.

because there is no time delay between the two steps. For this reason, V2C−V1O is eliminated based on rule 3 (see Figure A.5).

A computer algorithm has been developed to synthesize the diagnostic test procedure according to the aforementioned pruned search tree. The flowchart of this program is presented in Figure A.6. A copy of this code is also included in the Supporting Information—part I. The proper diagnostic test procedure for the present example can be determined with this program. In particular, this procedure consists of the following three steps: (1) open V-1 (V1O), (2) close V-2 (V2C), and then (3) wait for one unit of time. After completing these extra steps in the first scenario, valve V-1 will be opened and valve V-2 should remain open (because it is stuck at the open position). As a result, the observable liquid level is still unchanged at TH since both inlet and outlet flows are present. On the other hand, the liquid level in the second scenario must be lowered to TL eventually since both V-1 and V-2 can be correctly switched to their target positions.

Supporting Information Available: Program codes, data files, and Figures S1−S15. This material is available free of charge via the Internet at http://pubs.acs.org.

Literature Cited

(1) Venkatasubramanian, V.; Rengaswamy, R.; Yin, K.; Kavuri, S. N. A review of process fault detection and diagnosis, part i: Quantitative model based methods. *Comput. Chem. Eng.* **2003**, *27*, 293.

(2) Venkatasubramanian, V.; Rengaswamy, R.; Kavuri, S. N. A review of process fault detection and diagnosis, part ii: Qualitative model and search strategies. *Comput. Chem. Eng.* **2003**, *27*, 313.

(3) Venkatasubranmanian, V.; Rengaswamy, R.; Kavuri, S. N.; Yin, K. A review of process fault detection and diagnosis, part iii: Process history based methods. *Comput. Chem. Eng.* **2003**, *27*, 313.

(4) Nomikos, P.; MacGregor, J. F. Monitoring batch processes using multiway principal component analysis. *AIChE J.* **1994**, *40*, 1361.

(5) Nomikos, P.; MacGregor, J. F. Multivariate SPC charts for monitoring batch processes. *Technometrics* **1995**, *37*, 41.

(6) Kourti, T.; Macgregor, J. F. Process analysis, monitoring and diagnosis, using multivariate projection methods. *Chemom. Intell. Lab. Syst.* **1995**, *28* (1), 3.

(7) Kourti, T.; Nomikos, P.; Macgregor, J. F. Analysis monitoring and fault-diagnosis of batch processes using multiblock and multiway PLS. *J. Process Control* **1995**, *5* (4), 277.

(8) Undey, C.; Ertunc, S.; Cinar, A. Online batch fed-batch process performance monitoring, quality prediction, and variable contribution analysis for diagnosis. *Ind. Eng. Chem. Res.* **2003**, *42*, 4645.

(9) Lee, J. M.; Yoo, C. K.; Lee, I. B. Fault detection of batch processes using multiway kernel principal component analysis. *Comput. Chem. Eng.* **2004**, *28*, 1837.

(10) Ruiz, D.; Canton, J.; Nougues, J. M.; Espuna, A.; Puigjaner, L. On-line fault diagnosis system support for reactive scheduling in multipurpose batch chemical plants. *Comput. Chem. Eng.* **2001**, *25*, 829.

(11) Ruiz, D.; Nougues, J. M.; Calderon, Z.; Espuna, A.; Puigjaner, L. Neural network based framework for fault diagnosis in batch chemical plants. *Comput. Chem. Eng.* **2001**, *24* (2−7), 777.

(12) Pierri, F.; Paviglianiti, G.; Caccavale, F.; Mattei, M. Observer-based sensor fault detection and isolation for chemical batch reactors. *Eng. Appl. Artif. Intel.* **2008**, *21* (8), 1204.

(13) Chang, C. T.; Hwang, H. C. New development of the digraph-based techniques for fault-tree synthesis. *Ind. Eng. Chem.* **1992**, *31*, 1490.

(14) Andrews, J. D.; Morgan, J. M. Application of digraph method of fault tree construction to process plant. *Reliab. Eng.* **1986**, *14*, 85.

(15) Peterson, J. *Petri Net Theory and the Modeling of Systems*; Prentice-Hall: Englewood Cliffs, NJ, 1981.

(16) David, R.; Alla, H. Petri net for modeling of dynamic systems - a survey. *Automatica* **1994**, *30* (2), 175.

(17) Sampath, M.; Lafortune, S.; Sinnamohideen, K.; Teneketzis, D. Diagnosability of discrete-event systems. *IEEE Trans. Automatic Control* **1995**, *40* (9), 1555.

(18) Sampath, M.; Sengupta, R.; Lafortune, S.; Sinamohideen, K.; Teneketzis, D. Failure diagnosis using discrete-event models. *IEEE Trans. Control Syst. Technol.* **1996**, *4* (2), 105.

(19) Prock, J. A new technique for fault detection using Petri nets. *Automatica* **1991**, *27* (2), 239.

(20) Adamyan, A.; He, D. Failure and safety assessment of systems using Petri nets. *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, Washington, DC, 2002; p 1919.

(21) Adamyan, A.; He, D. Sequential failure analysis using counters of Petri net models. *IEEE Trans. Syst., Man, and Cybern.—Part A: Syst. Humans* **2003**, *33* (1), 1.

(22) Paoli, A.; Lafortune, S. Safe diagnosability of discrete event systems. *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, HI, 2003; p 2658.

(23) Chung, S. W. C.; Jeng, M. D. Failure diagnosis: a case study on modeling and analysis by Petri nets. *Procedings of the IEEE International Congerence on Systems, Man and Cybernetics*, 2003; p 2727.

(24) Ramirez-Trevino, A.; Beltran, E. R.; Rivera-Rangel, I.; Lopez-Mellado, E. Diagnosability of discrete event systems. A Petri net based approach. *Proceedings of the IEEE International Conference on Robotic and Automation*, 2004; p 541.

(25) Ushio, T.; Onishi, I.; Okuda, K. Fault detection based on Petri net models with faulty behaviors. *Proceedings of IEEE International Conference on System, Man, and Cybernetics, 11V14*, October 1998; p 113.

(26) Jiang, S.; Huang, Z.; Chandra, V.; Kumar, R. A polynomial algorithm for testing diagnosability of discrete-event systems. *IEEE Trans. Automatic Control* **2001**, *46* (8), 1318.

(27) Jiroveanu. G.; Boel, R.; Schutter, B. D. Fault diagnosis for time Petri nets. *Proceedings of the 8th Internation Workshop on Discrete Event Systems (WODES06)*, Ann Arbor, MI, 2006; p 313.

(28) Jiroveanu, G.; Schutter, B. D.; Boel, R. On-line diagnosis for time Petri nets. *Proceedings of the 17th International Workshop on Principles of Diagnosis (DX-06)*, Burgos, Spain, 2006; p 8.

(29) Chung, S.-L. Diagnosing PN-based models with partial observable transitions. *Int. J. Comp. Interated Manuf.* **2005**, *18* (2−3), 158.

(30) Cantu, M. *Delphi 2007Handbook*; self-printed and sold on www.Lulu.com.

Received for review June 9, 2009
Revised manuscript received February 26, 2010
Accepted March 4, 2010

IE901410P