Failure Diagnosis and Recovery based on DES Framework

Hyoung Il Son
Evaluation Technology Group
LG Production engineering Research Center
E-mail: chakurt@lge.com

Abstract—**As many industrial systems become complex, it is very difficult to identify the cause of failures. This paper presents a new failure diagnosis approach based on discrete-event systems (DES) framework. In particular, the approach is a hybrid of event-based and state-based ones leading to a simpler failure diagnoser with supervisory control capability. In our approach, we include the failure recovery events for failures in the system model in order to derive a diagnoser we refer to as a recoverable diagnoser. Further, in order to reduce the state size of the recoverable diagnoser, a procedure to construct a high-level diagnoser is presented.**

Index terms—**Failure diagnosis, failure recovery, discrete event systems, model reduction**

I. INTRODUCTION

Failure diagnosis in industrial system is a subject that received a great deal of attention in the past few decades. To solve diagnostic problems for large complex systems such as semiconductor manufacturing systems, automobile manufacturing systems, chemical processes, HVAC (Heating, Ventilation and Air Conditioning) units, and power plants, more systematic and efficient approaches are required because of the tremendous problem sizes.

In order to develop practical diagnostic systems, many theoretical frameworks have been proposed. These include fault tree analysis, analytical redundancy, expert systems, and model-based reasoning methods. Even though these techniques have their own merits, the real-world applications require some ways to circumvent individual limitations such as prohibitive computational burden, excessive sensitivity to modeling errors and sensor noise, and lack of systematic knowledge acquisition. Recently, DES methods [1]-[8] are recognized as one of the promising techniques because most industrial systems can be readily modeled as discrete event systems and DES technique can offer more systematic construction of a diagnostic system. Due to these advantages, there have been many approaches to the diagnosis problem by using DES technique [9]-[14].

Failure diagnosis system can be classified by two criteria. The first criterion is what is the state of a system when the diagnostic procedure is applied. In the off-line failure diagnosis, the system is assumed to be in an abnormal state when the failure diagnosis begins. The diagnostic system collects information from the failed system to draw inferences on the state of system and the cause of the failure. In contrast with the off-line failure diagnosis, the on-line failure diagnosis is applied while the system is in normal operation. The on-line diagnostic system collects information to determine whether the system is normal or not. If the system is in abnormal states, the diagnostic procedure tries to find the cause. The second criterion for classifying diagnosis system is whether the failure diagnoser actively intervenes with the system's operation. The passive failure diagnosis does not affect the system's operation. Instead, it simply observes the sequence of events and keeps the track of system states. On the other hand, the active failure diagnosis can change the system's operation by issuing a sequence of commands to determine the system's state and the cause of the failure [9], [12], [13].

The DES approach to the diagnosis problem is divided into an event-based approach and a state-based approach. In general, the event-based approach is simpler in its design procedure than the state-based approach. But its drawback is that the designed diagnoser has more states than the other. Also, unlike the state-based diagnosis, it does not require the knowledge on the state of the supervisor that controls the system's behavior.

This paper present an approach to design a passive on-line diagnoser based on the DES framework. Unlike the existing DES techniques, this approach is a hybrid of the event-based and the state-based approaches leading to a simpler failure diagnoser with supervisory control capability by taking recovery events for failures into consideration. By this extension, the diagnoser can allow the system to recover from a failure as well as detect and isolate the failure. Also, a procedure to construct a high-level diagnoser is presented in order to reduce the state size of the diagnoser.

This paper is organized into five sections. In the section following this introduction, a DES modeling technique and a design procedure for the proposed diagnoser are presented. In Section III, the concept of diagnosability along with its necessary and sufficient conditions is presented. In addition, the concept of recoverability that is first introduced by this paper is defined with its necessary and sufficient conditions. The methodology for reducing the state size of the recoverable diagnoser is presented in Section IV. Finally, we summarize the main contributions of this paper and outline the directions for future research in Section V.

II. RECOVERABLE DIAGNOSER DESIGN

In this section we will present the DES modeling procedure

0-7803-8599-3/04/$20.00 ©2004 IEEE

and the diagnoser design procedure based on the DES framework. While existing DES methods for failure diagnosis do not deal with the failure recovery problem, our approach takes into account failure recovery events in the DES modeling step. By this approach the diagnoser makes a DES return to the initial normal state by enabling a failure recovery event when the diagnoser detects a failure event.

A. DES Modeling

In general, we can assume that the DES to be diagnosed has a several system components including the plant components to be controlled and the supervisor for control action. First, let these plant components be modeled by the Finite State Automata (FSA)

$$\overline{G_i} = \left\{\overline{Q_i}, \overline{\Sigma_i}, \overline{\delta_i}, \overline{q_{0,i}}, \overline{Q_{m,i}}\right\}, \quad i = 1, ..., n \tag{1}$$

where $\overline{Q_i}$ is the state set, $\overline{\Sigma_i}$ is the event set, $\overline{\delta_i} : \overline{Q_i} \times \overline{\Sigma_i}^* \to \overline{Q_i}$ is the state transition function, $\overline{q_{0,i}}$ is the initial state, and $\overline{Q_{m,i}}$ is the marked state set that is a subset of the state set $\overline{Q_i}$. In defining transition function $\overline{\delta_i}$, the notation $\overline{\Sigma_i}^*$ means the set of sequences (strings) of events including the null event ε. To define the failure event of each component let us define the failure event set of a component as $\overline{\Sigma_{F,i}} = \left\{f_i^1, f_i^2, ..., f_i^m\right\}$. And define the failure recovery event set for failure events as $\overline{\Sigma_{RF,i}} = \left\{Rf_i^1, Rf_i^2, ..., Rf_i^n\right\}$, $n \le m$ (assuming that some failure recovery event can take care of more than one failure events). So the normal event set is defined as $\overline{\Sigma_{N,i}} = \overline{\Sigma_i} - \overline{\Sigma_{F,i}} - \overline{\Sigma_{RF,i}}$. As a result we partition the event set $\overline{\Sigma_i}$ into three disjoint event sets, i.e., the normal event set $\overline{\Sigma_{N,i}}$, the failure event set $\overline{\Sigma_{F,i}}$, and failure recovery event set $\overline{\Sigma_{RF,i}}$. That is, $\overline{\Sigma_i} = \overline{\Sigma_{N,i}} \dot{\cup} \overline{\Sigma_{F,i}} \dot{\cup} \overline{\Sigma_{RF,i}}$ where $\dot{\cup}$ denotes a disjoint union. Then, we can obtain the FSA of the total plant by the synchronous product of all FSAs $\overline{G_i}$. The resulting FSA is denoted by

$$\overline{G} = \left\{\overline{Q}, \overline{\Sigma}, \overline{\delta}, \overline{q_0}, \overline{Q_m}\right\} \tag{2}$$

where $\overline{Q}$, $\overline{\Sigma}$, $\overline{\delta}$, $\overline{q_0}$, and $\overline{Q_{m,i}}$ follow the previous definitions. In particular, the event set $\overline{\Sigma}$ can be divided into two disjoint sets, i.e., the controllable event set $\overline{\Sigma_c}$ and the uncontrollable event set $\overline{\Sigma_{uc}}$. And $\overline{\Sigma}$ can be also partitioned into the observable event set $\overline{\Sigma_o}$ and the unobservable event set $\overline{\Sigma_{uo}}$ that are also disjoint. Therefore, $\overline{\Sigma}$ can be written as $\overline{\Sigma} = \overline{\Sigma_c} \dot{\cup} \overline{\Sigma_{uc}} = \overline{\Sigma_o} \dot{\cup} \overline{\Sigma_{uo}}$.

As the supervisor for $\overline{G}$, (S, φ) can be used to generate the supremal controllable and observable sublanguage where S is an FSA, $S = \left\{X, \Sigma, \xi, x_0, X_m\right\}$ that can be obtained by the results in [6][7] and φ is a control map defined as $\varphi : X \to 2^{\Sigma}(\supseteq \Sigma_{uc})$. With $\overline{G}$ as the total plant and the supervisor (S, φ), the total system can be represented by a Finite State Moore Automaton (FSMA)

$$G = \left\{Q, \Sigma, \delta, q_0, Q_m, Y, \lambda, C, \gamma\right\}^1 \tag{3}$$

that is obtained by the meet product of plant FSA $\overline{G}$ and supervisor FSA S. In the FSMA G, $Q = \overline{Q} \times X$ is the state set; $\Sigma \subseteq \overline{\Sigma}$ is the event set; $\delta : Q \times \Sigma \to Q$ is the transition function; $q_0 = (\overline{q_0}, x_0)$ is the initial state with $\overline{q_0}$ and x_0 are the initial states of the plant and the supervisor, respectively; and Q_m is the marked state set that is a subset of the state set Q. Among the new components, Y is the sensor output set, $\lambda : Q \to Y$ is the sensor output map, $C \subseteq \Sigma_c$ is the control command set, and $\gamma : Q \to 2^C$ is the control command map. Here, the sensor output means the results of sensor measurements of the system while the control command set includes the event can be enabled by the supervisor.

B. Recoverable Diagnoser

As the first step to develop a recoverable diagnoser, the event set Σ is partitioned into the normal event set Σ_N, the failure event set Σ_F and the failure recovery event set Σ_{RF}[2]. That is, $\Sigma = \Sigma_N \dot{\cup} \Sigma_F \dot{\cup} \Sigma_{RF}$. And assume the failure event set Σ_F can be defined as $\Sigma_F = \left\{f_1, f_2, ..., f_m\right\}$ for a failure event $f_i, i = 1, 2, ..., m$. Denote the state q' that is reached from a certain state q of G by the failure event f_i as q_{f_i}: namely, q_{f_i} is defined as $q_{f_i} = q' = \delta(q, f_i)$. We define $F_1, F_2, ..., F_n$, $n \le m$ as failure modes, also $K = \left\{N, F_1, F_2, ... F_n\right\}, n \le m$ as state condition set of DES. Failure modes partition the failure event set Σ_F into n groups by aggregating some failure events that can be considered identical. In addition, assume that there is F_i-recovery event that is denoted by RF_i for each failure mode F_i. So we can define the failure recovery event set as $\Sigma_{RF} = \left\{RF_1, RF_2, ..., RF_n\right\}, n \le m$. Naturally all members of Σ_{RF} are controllable event, therefore $\Sigma_{RF} \subseteq \Sigma_c$.

---

[1] In general, $L(S/\overline{G}) = L(S)$ is true [1], [2], [6] so we can obtain G by just adding λ, Y to S.

[2] $\Sigma_N = \bigcup_i \overline{\Sigma_{N,i}}$, $\Sigma_F = \bigcup_i \overline{\Sigma_{F,i}}$, $\Sigma_{RF} = \bigcup_i \overline{\Sigma_{RF,i}}$

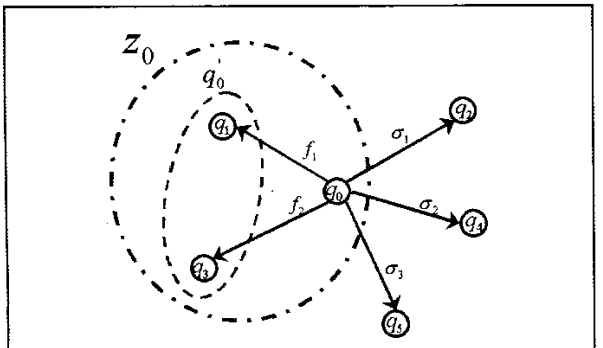

Fig. 1. Initial State of the Recoverable Diagnoser.

With the partitions on the event set, now let us define Q_N and Q_{F_i} by the following equations as disjoint partitions of the state set Q, i.e., $Q = Q_N \dot{\cup} Q_{F_1} \dot{\cup} Q_{F_2} \dot{\cup} \ldots \dot{\cup} Q_{F_n}$.

$$Q_N = q_0 \cup \left\{ q' \middle| \forall q \in Q_N, \forall \sigma \in \Sigma_N, q' = \delta(q,\sigma) \right\} \quad (4)$$

$$\begin{aligned} Q_{F_i} = & \left\{ q_{f_i} \middle| \forall q \in Q_N, \forall f_i \in F_i, q_{f_i} = \delta(q,f_i) \right\} \\ & \cup \left\{ q' \middle| \forall q \in Q_{F_i}, \forall \sigma \in \Sigma_N, q' = \delta(q,\sigma) \right\} \end{aligned} \quad (5)$$

Finally, by the state condition and disjoint state sets defined in equations (4) and (5), we define the state condition map $\kappa : Q \to \mathrm{K}$ as follows.

$$\kappa(q) = \begin{cases} N \text{ if } q \in Q_N \\ F_i \text{ if } q \in Q_{F_i} \end{cases} \quad (6)$$

Now let us define the recoverable diagnoser for DES G as

$$D = \left\{ Z, E, \varsigma, z_0, Z_m, \overline{\mathrm{K}}, \overline{\kappa} \right\} \quad (7)$$

where $Z = 2^Q - \phi$ is the state set, $E = Y \times C$ is the event set, z_0 is the initial state that is defined as $z_0 = \left\{ q_0 \cup q_0' \right\}$, $Z_m \supseteq Q_m$ is the marked state set, $\overline{\mathrm{K}} = 2^{\mathrm{K}} - \phi$ is the state condition set, $\overline{\kappa} : Z \to \overline{\mathrm{K}}$ is the state condition map. This type of diagnoser D is also referred to as *recoverable diagnostic supervisor* because it preserves the control map of the supervisor while performing diagnosis. For initial state definition, the notation q_0' is defined as follows and illustrated in Fig. 1.

$$q_0' = \bigcup_i \left\{ q' \middle| \forall f_i \in \Sigma_F, q' = \delta(q_0, f_i) \right\} \quad (8)$$

This definition allows the diagnoser to start from either a normal or a faulty state. This is very advantageous in the sense that there is no problem in operation of the diagnoser no matter when a failure occurs, i.e., before or after the initialization of diagnoser. Some previous works assumed that the system starts from a normal state [10]-[12], which may not be general enough for many applications. And in [13], the initial state is defined as all system states or all normal states. But this may be impractical in combining supervisory control and diagnosis because the supervisor has to start from some initial state.

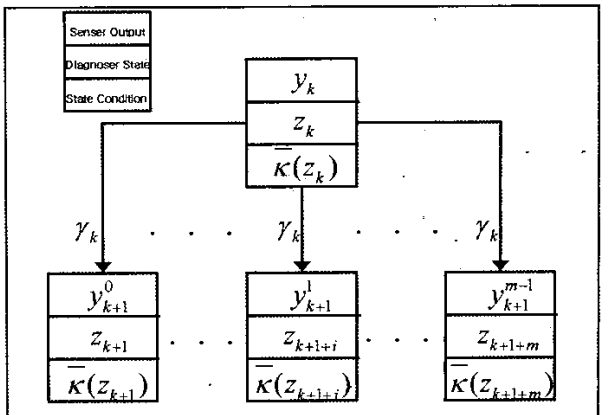

Fig. 2. Transition Function ς.

In order to design the diagnoser defined in (7) from FSMA G, we have to construct a new transition system excluding all unobservable events such as failure events and the observable events that are not the member of control command set of the supervisor. The next definition defines this new transition system.

Definition 1: Define the Control Command Transition Systems (CCTS) as all transition (q,σ,q') of DES G such that

$$\forall \sigma \in C, q' = \delta(q,\sigma).$$

CCTS contains only transitions that can be enabled by the supervisor. Physically, only events that are triggered by the control command from the supervisor are contained in CCTS. So, CCTS are the subset of transitions that are monitored by the supervisor.

We define the transition function ς of the diagnoser D as follows.

$$\varsigma(z_k, \gamma_k) = \begin{cases} z_{k+1} & \text{if } n(y_{k+1}) = 1 \\ z_{k+1+i} & \text{if } n(y_{k+1}) \geq 2, \\ \quad (0 \leq i \leq m-1) & \quad y_{k+1} = \left\{ y_{k+1}^0, \ldots, y_{k+1}^{m-1} \right\} \\ \text{undefined} & \text{elsewhere} \end{cases} \quad (9)$$

where $n(\bullet)$ means the number of $\bullet$.

The meaning of equation (9) can be explained by the following. If the supervisor issues a control command γ_k at the state z_k of diagnoser, then the state z_k transits to z_{k+1}. However, if there are more than one sensor outputs after the control command γ_k, we need to differentiate the next state z_{k+1} according to the sensor output. For this purpose, we define additional states as shown in Fig. 2.

Although this approach may look similar to [10]-[13], the transition function ς is different from the previous works and can be more efficient. This is because of the following two reasons. First, the diagnoser in [10]-[12] has to update the state condition whenever any observable events occur while this approach requires to update the state condition only when

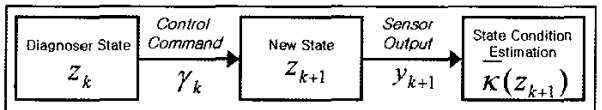

Fig. 3. On-line Diagnosis.

there are changes in the value of sensor output and the supervisor issues the new control command. So the number of states of the new diagnoser is at most the same as that of the diagnoser presented in [10]-[12]. The second reason is related to the fact that the state of the supervisor has to be included in the state output of the diagnoser presented in [13]. This requires the diagnoser to synchronize with the supervisor in order to know the state information of the supervisor. In contrast, the diagnoser in this research includes both diagnostic and control capabilities, which implies that the recoverable diagnostic supervisor needs no synchronization and is much simpler than the diagnoser presented in [13].

Figure 3 shows the schematic of on-line diagnosis for a DES. In Fig. 3, when the control command γ_k is issued at the present state z_k of the diagnoser, the state transits to a new state z_{k+1}. Then, the diagnoser reads the sensor output y_{k+1} of the present state z_{k+1}, and estimates the state condition of z_{k+1}.

In summary, the recoverable diagnoser design procedure is presented in the following.

Recoverable Diagnoser Design Procedure:

Step 1: Define the failure modes from the failure events.

Step 2: Define the failure recovery event set Σ_{RF}.

Step 3: Define the state condition map κ by using (4), (5), and (6).

Step 4: Find the initial state of diagnoser by using (8).

Step 5: Construct CCTS.

Step 6: Define the transition function ς by using CCTS and (9).

Step 7: Build the recoverable diagnoser by using the transition function. ■

III. Diagnosability and Recoverability

This section presents the definition of diagnosability and its necessary and sufficient conditions. In addition, the concept of recoverability is defined along with its necessary and sufficient conditions.

A. Diagnosability

In this section we explain how the diagnoser estimates the state condition of DES and detects the failure event. For this purpose, the state set Z of the diagnoser is classified into normal, F_i -uncertain, and F_i -certain states. The precise definitions are provided in the following.

Definition 2: The state z of the diagnoser is said to be normal if

$$\overline{\kappa(z)} = \{N\}$$

Definition 3: The state z of the diagnoser is said to be F_i -uncertain if

$$\overline{\kappa(z)} \supset \{F_i\}, \overline{\kappa(z)} \not\subset \{F_i\}$$

Definition 4: The state z of the diagnoser is said to be F_i -certain if

$$\overline{\kappa(z)} = \{F_i\}$$

Based on the above definitions, diagnosability of a diagnoser is defined by the following.

Definition 5: A diagnoser is F_i -diagnosable if the state of the diagnoser can be F_i -certain after the occurrence of at most a finite number of events, N_i, in the system, following both the initialization of the diagnoser and the occurrence of failure mode F_i. Also, if diagnoser is F_i -diagnosable for all failure modes F_i then the diagnoser is said to be diagnosable.

The physical meaning of diagnosability can be explained by the following. Suppose that a system looks normal to the diagnoser even after a failure mode F_i occurred. In that case, the state conditions of the diagnoser contain not only F_i but also N, i.e. the state of the diagnoser is F_i -uncertain. However, if the state conditions of the diagnoser contain only F_i without N after occurrences of some abnormal events, i.e. the state of the diagnoser is F_i -certain, the diagnoser can determine that the failure mode F_i has occurred without any ambiguity.

Before we state the theorem for diagnosability, let us define a cycle and an indeterminate cycle.

Definition 6: A set of states $\{q_1, q_2, ..., q_{n+1}\}$ is said to form a cycle if $q_{i+1} = \delta(q_i, \sigma_i)$, $i = 1, 2, ..., n$ and $q_1 = \delta(q_{n+1}, \sigma_{n+1})$.

Definition 7: Assume that a set of states $\{z_1, z_2, ..., z_{n+1}\}$ in the diagnoser that is F_i -uncertain forms a cycle. Then the set of states $\{z_1, z_2, ..., z_{n+1}\}$ is said to form an F_i -indeterminate cycle if there exist cycles $\{q_1^N, q_2^N, ..., q_{k+1}^N\}$ and $\{q_1^{F_l}, q_2^{F_l}, ..., q_{l+1}^{F_l}\}$ that satisfy the following.

1. $\{q_1^N, q_2^N, ..., q_{k+1}^N\}$ forms a cycle such that $\kappa(q_m^N) = N$, $q_m^N \in z_r$, $m = 1, 2, ..., k+1$, $r = 1, 2, ..., n+1$ and $\{q_1^N, q_2^N, ..., q_{k+1}^N\} \subseteq \{z_1, z_2, ..., z_{n+1}\}$. This cycle said to be N -cycle; and
2. $\{q_1^{F_i}, q_2^{F_i}, ..., q_{l+1}^{F_i}\}$ *forms a cycle such that* $\kappa(q_m^{F_i}) = F_i$, $q_m^{F_i} \in z_r$, $m' = 1, 2, ..., l+1$, $r = 1, 2, ..., n+1$ and $\{q_1^{F_i}, q_2^{F_i}, ..., q_{l+1}^{F_i}\} \subseteq \{z_1, z_2, ..., z_{n+1}\}$. *This cycle said to be* F_i *-cycle.*

Theorem 1: The diagnoser is F_i-diagnosable if and only if there is no F_i-indeterminate cycle in the diagnoser for the failure mode F_i.

Proof: After the occurrence of the failure mode F_i, the state of the diagnoser will be one of the following three types: normal, F_i-uncertain and F_i-certain states.

1. In case of normal state condition.

(Sufficiency) Because there is no F_i-indeterminate cycle, there exists a path (sequence or string) to a state whose state condition is F_i. Therefore, the state of the diagnoser eventually becomes F_i-certain, which implies that the diagnoser is F_i-diagnosable.

(Necessity) Because the diagnoser is F_i-diagnosable, there exist the path to the state that is F_i-certain in the diagnoser. So there is no F_i-indeterminate cycle formed by F_i-uncertain states.

2. In case of F_i-uncertain state condition.

Sufficiency and Necessity are satisfied by the same reasoning as in the case of normal state condition.

3. In the case of F_i-certain.

Because the present state of the diagnoser is F_i-certain, the diagnoser is F_i-diagnosable. Since there is no F_i-indeterminate cycle, there exist a path to the state that is F_i-certain. ■

B. Recoverability

In this section we define the recoverability of a recoverable diagnoser and present the necessary and sufficient condition for recoverability. First, the recoverability is defined in Definition 8 followed by the necessary and sufficient conditions for recoverability in Theorem 2.

Definition 8: The recoverable diagnoser is F_i-recoverable if

$$\forall\, z_R \text{ such that } \overline{\kappa(z_R)} = \{F_i\},\ RF_i \in \gamma_R(z_R).$$

In other words, if the diagnoser can enable the failure recovery event RF_i at all F_i-certain states, then the diagnoser is said to be F_i-recoverable.

Theorem 2: The recoverable diagnoser is F_i-recoverable if and only if

Condition 1. The failure recovery event RF_i can be enabled at any state q_R of which state condition is F_i, and

Condition 2. There is no F_i-indeterminate cycle in the recoverable diagnoser.

Proof: (Sufficiency) By the condition 2, the recoverable diagnoser satisfies Theorem 1 so the recoverable diagnoser is F_i-diagnosable. Therefore, there exists a state z_R in the recoverable diagnoser that is F_i-certain. And by the condition 1, at any state z_R that is F_i-certain, it is true that $RF_i \in \gamma_R(z_R)$. Therefore the recoverable diagnoser satisfies the Definition 8 and is F_i-recoverable.

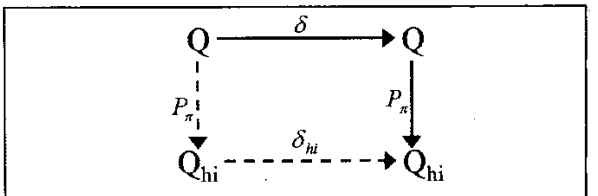

Fig. 4. High-level Projection P_π of DES.

(Necessity) Because the recoverable diagnoser is F_i-recoverable it is true that $RF_i \in \gamma_R(z_R)$ at all state z_R that is F_i-certain by Definition 8. Therefore, the failure recovery event RF_i can be enabled at all state $q_R (q_R \in z_R)$ of the DES, and also because z_R is F_i-certain, $\kappa_R(q_R) = F_i$. Then, because the recoverable diagnoser is also F_i-recoverable, there exists a state z_R that is F_i-certain. Therefore, there is no F_i-indeterminate cycle in the recoverable diagnoser. ■

IV. High-level Diagnoser Design

Even though the recoverable diagnoser developed in Section III can identify the cause of failure and recover from it, it may be very difficult to design such a diagnoser for large-scale systems commonly found in industrial applications. In order to reduce the size of the diagnoser, this section applies the theory of hierarchical control to the diagnosis problem. We present the model reduction scheme of DES to make a high-level diagnoser along with the proof of the equivalency of the original recoverable diagnoser and the high-level diagnoser.

A. Model Reduction

The basic idea of model reduction scheme is to partition states of DES as equivalence classes of sensor output map, control command map, and state condition map because the diagnoser transits and estimates the state only by sensor output, control command, and state condition. So, those states in an equivalence class can be treated identically for the purpose of supervisory control and failure diagnosis.

First, we define the high-level DES.

Definition 9: Define the high-level DES as the DES that is reconstructed by the coarsest partition π such that

$$\pi \le \ker\lambda \wedge \ker\gamma \wedge \ker\kappa$$ [3]

The partition π is the coarsest partition that preserves the

[3] Notation $\ker\lambda$ means the coset (equivalent class) for the binary relation λ.

information on partitions $\ker\lambda$, $\ker\gamma$, and $\ker\kappa$. Also, the high-level projection P_π is defined as $P_\pi : Q \to Q_{hi}$ for the partition π that is illustrated in Fig. 4.

The states of a DES are aggregated by high-level projection P_π into a high-level state that has the same sensor output, control command, and state condition information.

Define the high-level DES based on Definition 9 as FSMA

$$G_{hi} = \left\{Q_{hi}, \Sigma_{hi}, \delta_{hi}, q_{hi,0}, Q_{hi,m}, Y_{hi}, \lambda_{hi}, C_{hi}, \gamma_{hi}\right\} \quad (10)$$

where G_{hi}, Σ_{hi}, δ_{hi}, $q_{hi,0}$, and $Q_{hi,m}$ are the state set, the event set, the transition function, the initial state, and the marked state set, respectively; and Y_{hi}, λ_{hi}, C_{hi}, and γ_{hi} are the sensor output set, the sensor output map, the control command set, and the control command map, respectively.

B. High-level Diagnoser

Let the DES

$$D_{hi} = \left\{Z_{hi}, E_{hi}, \varsigma_{hi}, z_{hi,0}, Z_{hi,m}, \overline{\mathrm{K}_{hi}}, \overline{\kappa_{hi}}\right\} \quad (11)$$

denote the high-level diagnoser for the high-level DES. For convenience, we refer to the high-level recoverable diagnoser as the high-level diagnoser hereinafter. In (11), Z_{hi}, E_{hi}, ς_{hi}, $z_{hi,o}$, and $Z_{hi,m}$ are the state set, the event set, the transition function, the initial state and the marked state set, respectively. And $\overline{\mathrm{K}_{hi}}$ and $\overline{\kappa_{hi}}$ are the state condition set and state condition map, respectively.

We can also define the diagnosability and recoverability for high-level diagnoser as given in Definitions 5 and 8, respectively. And the necessary and sufficient condition for diagnosability and recoverability are also the same as proven in Theorem 1 and 2.

C. Equivalency

We show that the high-level diagnoser is equivalent to the original recoverable diagnoser in the following.

Theorem 3: The recoverable diagnoser D and the high-level diagnoser D_{hi} are equivalent.

Proof: We have to show that the state conditions of D and D_{hi} are identical for the state pair (q, q_{hi}) that has the same sensor output and control command. That is, we have to show the following equation holds.

$$\left[\forall\lambda(q) = \lambda_{hi}(q_{hi}) \wedge \gamma(q) = \gamma_{hi}(q_{hi})\right], \overline{\kappa(z)} = \overline{\kappa_{hi}(z_{hi})}$$

First, because $P_\pi(q_k) = q_{hi,k}$ and $\pi \le \ker\lambda \wedge \ker\gamma$, for z_k such that $q_k \in z_k$ and $z_{hi,k}$ such that $q_{hi,k} \in z_{hi,k}$, it follows that $P_\pi(z_k) = z_{hi,k}$ and $\overline{\kappa_{hi}(z_{hi,k})} = \overline{\kappa_{hi}\{P_\pi(z_k)\}}$. Then, from $\pi \le \ker\kappa$ it follows that $\overline{\kappa_{hi}\{P_\pi(z_k)\}} = \overline{\kappa(z_k)}$. Therefore, $\overline{\kappa(z_{hi,k})} = \overline{\kappa(z_k)}$.

V. Conclusions

This paper presents a new approach for on-line passive diagnoser that is capable of not only the failure diagnosis but also the supervisory control. This new approach is a hybrid of event-based and state-based strategies along with the introduction of failure recovery events and hierarchical control concept. The contributions of the paper can be summarized as follows.

This paper establishes a new failure diagnosis approach based on the combination of two well-known approaches, i.e., event-based and state-based. Compared to the event-based approach, the new approach can construct a simpler diagnoser because there is no need to update the state condition for all observable events. In comparison with the state-based approach, the hybrid approach allows much simpler implementation because there is no need to synchronize the diagnoser with the supervisor of the system.

By introducing failure recovery events, the supervisory controller and failure diagnoser can be integrated so that an action can be taken for failure recovery once the failure is diagnosed.

In order to apply this approach to real-world problems where the system quickly becomes too large to be handled, the theory of hierarchical control is applied to the diagnoser design. The high-level diagnoser performs exactly identical functions of the original diagnoser with less number of states.

References

[1] P. J. Ramadge and W. M. Wonham, "The control of discrete event systems," Proc. IEEE, vol. 77, pp. 81-98, Jan. 1989.

[2] R. Kumar and V. Garg, Modeling and Control of Logical Discrete Event Systems, Kluwer Academic Publishers.

[3] C. G. Cassandras, Discrete Event Systems: Modeling and Performance Analysis, Richard D. Irwin, Inc.

[4] J. E. Hopcroft and J. D. Ullman, Introduction to Automata Theory, Languages and Computation, Addision-Wesley, 1979.

[5] B. A. Davey and H. A. Priestley, Introduction to Lattices and Order, Cambridge University Press, 1990.

[6] W. M. Wonham, Notes on Control of Discrete Event Systems, Department of Electrical and Computer Engineering, University of Toronto, 1998.

[7] R. D. Brandt, V. Garg, R. Kumar, F. Lin, S. I. Marcus and W. M. Wonham, "Formulas for calculating supremal controllable and normal sublanguages," Syst. Contr. Lett., vol. 15, no. 2, pp. 111-117, Aug. 1990.

[8] W. M. Wonham and P. J. Ramadge, "On the supremal controllable sublanguage of a given language," SIAM J. Control and Optimization, Vol. 25, No. 3, May, 1987.

[9] F. Lin and T. W. Lin, "Diagnosability of Discrete Event Systems and Its Applications to Circuit Testing," Proceedings of the 36th Midwest Symposium on Circuits and Systems, 1993.

[10] Meera Sampath, A Discrete Event Systems Approach to Failure Diagnosis, Ph.D. Thesis, The University of Michigan, December, 1995.

[11] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Diagnosability of Discrete-Event Systems," IEEE Trans. Automat. Contr., vol. 40, no. 9, pp. 1555-1575, Sept., 1995.

[12] M. Sampath, S. Lafortune, and D. Teneketzis, "Active Diagnosis of Discrete-Event Systems," IEEE Trans. Automat. Contr., vol. 43, no. 7, pp. 908-929, July, 1998.

[13] Shahin Hashtrudi Zad, Fault Diagnosis in Discrete-Event and Hybrid Systems, Ph.D. Thesis, The University of Toronto, 1999.

[14] Yongseok Park, Model-based Monitoring of Discrete Event Systems, Ph.D. Thesis, The Purdue University, May, 1996.