Chapter from the book *Advances in Robotics, Automation and Control*

Downloaded from:
http://www.intechopen.com/books/advances_in_robotics_automation_and_control

Interested in publishing with IntechOpen?
Contact us at book.department@intechopen.com
Fault Diagnosis in Discrete Event Systems using Interpreted Petri Nets

Jesús Arámburo-Lizárraga, Antonio Ramírez-Treviño, Ernesto López-Mellado and Elvia Ruiz-Beltrán
CINVESTAV Unidad Guadalajara México

1. Introduction

Diagnosability property and fault detection schemes have been widely addressed on centralized approaches using the global model of the Discrete Event System (DES). Roughly speaking, diagnosability is the property of determining if using the system model is possible to detect and locate the faulty states in a finite number of steps. In the works (Sampath, et al., 1995) and (Sampath, et al., 1996), a method for modeling a DES using finite automata is proposed; based on this model, a diagnoser is derived. The cycles in the diagnoser are used to determine when the DES is diagnosable.

Recently, fault diagnosis of DES has been addressed through a distributed approach allowing breaking down the complexity when dealing with large and complex systems (Benveniste, et al., 2003; O. Contant, et al., 2004; Debouk, et al., 2000; Genc & Lafortune, 2003; Jiroveanu & Boel, 2003; Pencolé, 2004; Arámburo-Lizárraga, et al., 2005).

In (Debouk, et al., 2000) it is proposed a decentralized and modular approach to perform failure diagnosis based on Sampath’s results (Sampath, et al., 1995). In (Contant, et al., 2004) and (Pencolé, 2004) the authors presented incremental algorithms to perform diagnosability analysis based on (Sampath, et al., 1995) in a distributed way; they consider systems whose components evolve by the occurrence of events; the parallel composition leads to a complete system model intractable. In (Genc & Lafortune, 2003) it is proposed a method that handles the reachability graph of the PN model in order to perform the analysis similarly to (Sampath, et al., 1995); based on design considerations the model is partitioned into two labelled PN and it is proven that the distributed diagnosis is equivalent to the centralized diagnosis; later, (Genc & Lafortune, 2005) extend the results to systems modeled by several labelled PN that share places, and present an algorithm to determine distributed diagnosis.

In (Qiu & Kumar, 2005) it is studied the codiagnosability property, this property guarantees that any faults occurred in the system must be detected by at least one local diagnoser in a finite number of steps using the local information, besides, a notion of safe-codiagnosability is mentioned to capture the fact that the system has a safe specification while the system performance is tolerable. (Arámburo-Lizárraga, et al., 2005) proposes a methodology for designing reduced diagnosers and presents an algorithm to split a global model into a set of communicating sub-models for building distributed diagnosers. The diagnosers handle a system sub-model and every diagnoser has a set of communication events for detecting and
locating the faults of the corresponding sub-model. (Arámburo-Lizárraga, et al., 2007) shows how to design low interaction distributed diagnosers reducing the communication among them and proposes a redundant distributed diagnoser scheme composed by a set of independent modules handling two kinds of redundancy (duplication or TMR).

This work considers the system modeled as an interpreted PN (IPN) allowing describing the system with partially observable states and events; the model includes the possible faults it may occur. In order to build such a model, this work presents a bottom-up modeling methodology in which the behavior of the system elements is decomposed into state variables; a range for each state variable must be settled. These ranges represent the possible values of state variables. Afterwards, these ranges are coded into IPN (modules), where each value is represented by a different place. Then two composition operators for joining modules are used; the first one is named synchronic composition, which merges transitions according to certain rules. It is similar to the synchronous product presented in (Giua & DiCesare, 1994); the second one is named permissive composition, which uses selfloops for enabling transitions among modules; this allows constraining the model behavior into the actual system behavior. This new operator avoids the use of tuning phases of other modeling methods used in supervisory control (Giua & DiCesare, 1994). Based on the derived IPN model with the proposed methodology, the diagnosability property for IPN models is introduced. Roughly speaking, this property says that an IPN is diagnosable if it is possible to know both, when a faulty place is marked and which faulty place is marked. This property is closely related to the observability property (Aguirre-Salas, et al., 2002) and (Ramírez-Treviño, et al., 2003) and polynomial algorithms to test when an IPN is diagnosable are derived, avoiding the reachability analysis of other approaches. Also, a distributed diagnoser is presented; every distributed diagnoser uses the local information or communication among diagnosers for detecting and locating a system fault. The diagnosability property is preserved in the distributed architecture. Redundancy techniques could be applied to the distributed diagnosers to detect and locate a malfunction in the distributed diagnosers set.

The chapter is organized as follows: section 2 provides basic definitions of PN, IPN and the modeling methodology are presented. In section 3 the property of input-output diagnosability is defined and characterized, a diagnoser scheme devoted to detect and isolate failure states is also presented. Section 4 presents a procedure to build a reduced IPN model. Section 5 describes a method for model decomposition allowing interaction distributed diagnosers, also, it is presented a redundant scheme for reliable diagnosis applying redundancy to the distributed diagnosers. Finally, conclusions are given.

2. Basic notations and system modeling

2.1 Petri net basics

We consider systems modeled by Petri Nets and Interpreted Petri Nets. A Petri Net structure is a graph \( G = (P, T, I, O) \) where: \( P = \{p_1, p_2, \ldots, p_n\} \) and \( T = \{t_1, t_2, \ldots, t_m\} \) are finite sets of nodes called respectively places and transitions, \( I (O): P \times T \rightarrow \mathbb{Z}^+ \) is a function representing the weighted arcs going from places to transitions (transitions to places), where \( \mathbb{Z}^+ \) is the set of nonnegative integers.

The symbol \( \cdot t_i (P) \) denotes the set of all places \( p_i \) such that \( I(p_i, t_i) \neq 0 \) \( (O(p_i, t_i) \neq 0) \). Analogously, \( \cdot p_i (T) \) denotes the set of all transitions \( t_j \) such that \( O(p_i, t_j) \neq 0 \) \( (I(p_i, t_j) \neq 0) \) and the incidence matrix of \( G \) is \( C = [c_{ij}] \), where \( c_{ij} = O(p_i, t_j) - I(p_i, t_j) \).
A marking function $M: P \rightarrow \mathbb{Z}^+$ represents the number of tokens (depicted as dots) residing inside each place. The marking of a PN is usually expressed as an $n$-entry vector. A Petri Net system or Petri Net (PN) is the pair $N=(G,M_0)$, where $G$ is a PN structure and $M_0$ is an initial token distribution. $R(G,M_0)$ is the set of all possible reachable markings from $M_0$ firing only enabled transitions. In a PN system, a transition $t_i$ is enabled at marking $M_k$ if $\forall p_i \in P$, $M_k(p_i) \geq l(p_i,t_i)$; an enabled transition $t_i$ can be fired reaching a new marking $M_{k+1}$ which can be computed as $M_{k+1} = M_k + Cv_k$, where $v_k(i)=0, \neq j, v_k(j)=1$.

Interpreted Petri Nets (IPN) (Ramírez-Treviño, et al., 2003) is an extension to PN that allow to associate input and output signals to PN models. An IPN $(Q, M_0)$ is an Interpreted Petri Net structure where $Q = (G, \Sigma, \lambda, \varphi)$ with an initial marking $M_0$, $G$ is a PN structure, $\Sigma = \{a_1, a_2, ... , a_t\}$ is the input alphabet of the net, where $a_i$ is an input symbol, $\lambda: T \rightarrow \Sigma \cup \{\varepsilon\}$ is a transition labelling function with the following constraint: $\forall t_j, t_k \in T, j \neq k$, if $\forall p_i, I(p_i,t_j) = I(p_i,t_k) \neq 0$ and both $\lambda(t_j) \neq \varepsilon$, $\lambda(t_k) \neq \varepsilon$, then $\lambda(t_j) \neq \lambda(t_k)$; $\varepsilon$ represents an internal system event, and $\varphi: R(G,M_0) \rightarrow (\mathbb{Z}^+)q$ is an output function that associates to each marking an output vector, where $q$ is the number of outputs. In this work $\varphi$ is a $q \times n$ matrix. If the output symbol $i$ is present (turned on) every time that $M(p_i) \geq 1$, then $\varphi(i,j)=1$, otherwise $\varphi(i,j)=0$.

A transition $t_j \in T$ of an IPN is enabled at marking $M_k$ if $\forall p_i \in P$, $M_k(p_i) \geq l(p_i,t_j)$. If $t_j$ is enabled at marking $M_k$ and $\lambda(t_j)$ is present, then $t_j$ can be fired reaching $M_{k+1}$, i.e., $M_k \xrightarrow{t_j} M_{k+1}$; $M_{k+1}$ can be computed using the state equation:

$$M_{k+1} = M_k + Cv_k$$  
$$y_k = \varphi(M_k)$$

where $C$ and $v_k$ are defined as in PN and $y_k \in (\mathbb{Z}^+)q$ is the k-th output vector of the IPN.

The sequence $\sigma = t_1 \ldots t_k$ is a firing transition sequence of an IPN $(Q,M_0)$ if $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \ldots \ldots \xrightarrow{t_k} M_k$. According to functions $\lambda$ and $\varphi$, transitions and places of an IPN $(Q,M_0)$ are classified. If $\lambda(t_i) \neq \varepsilon$ the transition $t_i$ is said to be manipulated. Otherwise it is non-manipulated. A place $p_i \in P$ is said to be measurable if the i-th column vector of $\varphi$ is not null, i.e. $\varphi(\bullet,i) \neq 0$. Otherwise it is non-measurable. The following concepts are useful in the study of the diagnosability property.

The set $\mathcal{E}(Q,M_0) = \{ \sigma = t_1 \ldots t_k ... \wedge M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \ldots \ldots \xrightarrow{t_k} M_k \} \text{ of all firing transition sequences }$ is called the firing language of $(Q,M_0)$. A sequence of input-output symbols of $(Q,M_0)$ is a sequence $\omega = (a_0,y_0)(a_1,y_1)...(a_n,y_n)$, where $a_i \in \Sigma \cup \{\varepsilon\}$. The symbol $a_{i+1}$ is the current IPN input when the output changes from $y_i$ to $y_{i+1}$. It is assumed that $a_0 = \varepsilon$ and $y_0 = \varphi(M_0)$.

The firing transition sequence $\sigma \in \mathcal{E}(Q,M_0)$ whose firing actually generates $\omega$ is denoted by $\alpha$. The set of all possible firing transition sequences that could generate the word $\omega$ is defined as $\Omega(\omega) = \{ \sigma \mid \sigma \in \mathcal{E}(Q,M_0) \land \text{the firing of } \sigma \text{ produces } \omega \}$. The set $\Lambda(Q,M_0)$ is $\{ \omega \mid \omega \text{ is a sequence of input-output symbols} \}$ denotes the set of all sequences of input-output symbols of $(Q,M_0)$ and the set of all input-output sequences of length greater or equal than $k$ will be denoted by $\Lambda^k(Q,M_0)$, i.e. $\Lambda^k(Q,M_0) = \{ \omega \in \Lambda(Q,M_0) \mid |\omega| \geq k \}$ where $k \in \mathbb{N}$.
The set $\Lambda_0(Q,M_0) = \{ \omega \in \Lambda(Q,M_0) | \sigma \in \Omega(\omega) \text{ such that } M_\omega \xrightarrow{\sigma} M_j \text{ and } M_j \text{ enables no transition, or when } M_j \xrightarrow{\sigma} \text{ then } C(\bullet,t) = 0 \} \}$ denotes all input-output sequences leading to an ending marking in the IPN (markings enabling no transition or only self-loop transitions).

An IPN $(Q,M_0)$ described by the state equation (1) is event-detectable iff the firing of any pair of transition $t_i, t_j$ of $(Q,M_0)$ can be distinguished from each other by the observation of the sequences of input-output symbols.

The following lemma (Rivera-Rangel, et al., 2005) gives a polynomial characterisation of event-detectable IPN.

**Lemma 1**: A live IPN given by $(Q,M_0)$ is event-detectable if and only if:
1. $\forall t_i,t_j \in T$ such that $\lambda(t_i) = \lambda(t_j)$ or $\lambda(t_i) = \epsilon$ it holds that $\varphi_C(\bullet,t_i) = \varphi_C(\bullet,t_j)$ and
2. $\forall t_k \in T$ it holds that $\varphi_C(\bullet,t_k) \neq 0$.

### 2.2 System modeling

We work with the modeling methodology proposed in (Ramírez-Treviño, et al., 2007). The methodology follows a modular bottom-up strategy. After identifying the system components, a set of state variables is assigned to every component, each state variable behavior is modeled by a PN model, herein named module. Then the set of modules are integrated into a single model according to the appropriate relationships achieved through two module composition operations. This methodology builds binary IPN modules to represent the behavior of each component of the identified DES and the relationships between them. The model captures the normal and faulty behavior of the individual components of the system. Below it is described the detailed steps of the methodology.

#### 2.2.1 Modeling methodology

1. **System components.** - The system components must be identified and named. Afterwards, a finite set $\text{System Components} = \{sc_1, sc_2, ..., sc_n\}$ of these names must be created. A system component could be a valve, a motor, a system resource, etc.

2. **State variables.** - For each system component, the different variables needed to represent its behavior must be chosen. In other words, the finite set $\text{State Variables}_i = \{sv^{ij}_1, sv^{ij}_2, ..., sv^{ij}_m\}$ associated to the system component $sc_i \in \text{System Components}$ must be built. These variables could represent the position (for instance a valve position), velocity, voltage, etc. of each system component, or could represent a task descriptor (for instance a machine state). There exists at least one state variable for each system component.

3. **Set of values.** - For each state variable $sv^{ij}_j \in \text{State Variables}_i$, the set $\text{Value}_{sv^{ij}_j} = \{val^{ij}_1, val^{ij}_2, ..., val^{ij}_r\}$ of possible values of $sv^{ij}_j$ must be stated. Necessary faulty values should also be considered in this set. For instance, the variable "valve_position" may take four values: "Open", "Closed", "ErrorOnOpen" and "ErrorOnClosed", or the variable "task_machine1" may take three values: "loading", "processing", and "unloading".

4. **Codification.** - The values in each set $\text{Value}_{sv^{ij}_j}$ must be represented in terms of PN markings. This can be easily achieved if binary places are used. Thus, for each $sv^{ij}_j \in$...
State Variables $i$ a set $P_{i} \equiv \{p_{i1}^{u}, p_{i2}^{u}, ..., p_{in}^{u}\}$ of places such that $|Value_{i}^{u}| = |P_{i}|$ must be created. The marking of these places is binary and mutually exclusive. Then, $M(p_{i}) = 1$ means that the variable $sv_{i}^{u}$ takes the value $val_{i}^{u}$. Because of the existence of faulty values, the set of places can be partitioned into the subsets $P_{i}^{F}$ and $P_{i}^{N}$, representing the faulty and normal values respectively.

5. Event modeling.- For each pair of values $val_{m}^{u}$, $val_{n}^{u}$ such that the state variable $sv_{i}^{u}$ could change from value $val_{m}^{u}$ to a value $val_{n}^{u}$, a transition $t_{mn}^{u}$ must be created. Then, one arc going from place $p_{m}^{u}$ to transition $t_{mn}^{u}$ and one arc going from transition $t_{mn}^{u}$ to place $p_{n}^{u}$ must be added.

6. Initial marking.- The initial marking is defined as: $M_{0}(p_{m}^{u}) = 1$ if the initial value of the variable $sv_{i}^{u}$ is $val_{m}^{u}$ and $M_{0}(p_{m}^{u}) = 0$ otherwise.

7. Output.- The output of this algorithm is a set of isolated PN modules, each one modeling the behavior of a state variable $sv_{i}^{u}$.

8. Perform synchronous composition and permissive among modules IPN to obtain the global IPN model system (see Alcaraz-Mejía et al. 2003) and (Ramírez-Treviño et al. 2007).

In the IPN system model the sets of nodes are partitioned into faulty nodes ($P^{F}$, places coding faulty states, and $T^{F}$, transitions leading to faulty states) and normal functioning nodes ($P^{N}$ and $T^{N}$); so $P = P^{F} \cup P^{N}$ and $T = T^{F} \cup T^{N}$. $p_{i}^{N}$ denotes a place in $P^{N}$. Since $P^{N} \subseteq P$ then $p_{i}^{N}$ also belongs to $(Q, M_{0})$. The set of risky places of $(Q, M_{0})$ is $P^{R} = \ast T^{F}$. The post-risk transition set of $(Q, M_{0})$ is $T^{R} = P^{R} \cap T^{N}$. $(Q^{N}, M_{0}^{N})$ denotes the embedded normal behavior of $(Q, M_{0})$, i.e., $(Q^{N}, M_{0}^{N})$ is the subnet induced by normal nodes.

**Example 1.** Consider the producer-consumer scheme depicted in figure 1.

![Producer-Consumer with buffer of 2-slots scheme](image-url)
CU stops its consumption. The places $p_1$, $p_2$, $p_3$ represent the normal PU behavior and $p_{11}$ represents the faulty behavior. Places $p_4$, $p_5$, $p_6$ represent the normal CU behavior and $p_{12}$ represents the faulty behavior. The places $p_7$, $p_8$, $p_9$ and $p_{10}$ represent the 2-slots of the buffer. Function $\lambda$ is defined as $\lambda(t_1)=a$, $\lambda(t_5)=b$ and $\lambda(t_i)=\varepsilon$ for others transitions. Measurable places are $p_3$, $p_6$, $p_8$, $p_{10}$, $P^R = \{p_3, p_6\}$, $T^R = \{t_1, t_8\}$, $T^F = \{t_9, t_{10}\}$ and $P^F = \{p_{11}, p_{12}\}$.

3. Centralized diagnosability

The characterisation of input-output diagnosable IPN is based on the partition of $R(Q,M_0)$ into normal and faulty markings where all the faulty markings must be distinguishable from other reachable markings.

![Fig. 2. IPN modules of the identified components](image1)

![Fig. 3. Normal behavior of the IPN of the producer-consumer scheme](image2)

![Fig. 4. Normal and Faulty behavior of the IPN of the producer-consumer scheme](image3)
Definition 1: An IPN given by \((Q,M_0)\) is said to be input-output diagnosable in \(k < \infty\) steps if any marking \(M_t \in F\) is distinguishable from any other \(M_k \in R(Q,M_0)\) using any word \(\omega \in \Lambda^*(Q,M_k) \cup \Lambda_0(Q,M_0), \) \(F = \{M | \exists p_k \in P^f \text{ such that } M(p_k) > 0, M \in R(Q,M_0)\}\).

The following result provides sufficient structural conditions for determining the input-output diagnosability of an IPN model.

Theorem 1: Let \((Q,M_0)\) be a binary IPN, such that \((Q^N,M_0^N)\) is live, strongly connected and event detectable. Let \([X_1,...,X_t]\) be the set of all T-semiflows of \((Q,M_0)\). If \(\forall p_i^N \in P^N, (p_i^N)^* \cap T^f \neq \emptyset\) the following conditions hold:

1. \(\forall t, \exists j, X_t(j) \geq 1, \text{ where } t_i \in (p_i^N)^* - T^f,\)
2. \(\forall t, \exists (p_i^N)^* - T^f, *\left(t_i\right) = \{p_i^N\} \text{ and } \lambda(t_i) \neq e.\)

Then the IPN \((Q,M_0)\) is input-output diagnosable.

Proof: Assume that \((Q,M_0)\) meets all conditions of the theorem. Since \((Q^N,M_0^N)\) is live, then it is live by places (see Desel & Esparza, 1995), i.e. \(\forall p \in P\) there exists a marking \(M_k\) such that \(M_k(p) > 0\), previous observation is also valid for any \(p_i^N \in P^N, (p_i^N)^* \cap T^f \neq \emptyset.\) Thus there exists \(\omega_a\) such that \(M_0 \xrightarrow{\alpha} M_k\) and \(M_k(p_i^N) = 1.\) The input-output symbol word generating \(\omega_a\) will be denoted by \(\omega.\) Choose the longest transition firing sequence \(\omega_a\) not including transitions in \(T^f\) (could be the empty one). Such sequence must be marking a place \(p_i^N \in P^N, (p_i^N)^* \cap T^f \neq \emptyset\) since in the next step a faulty transition should be fired.

Since the initial marking is known, \((Q^N,M_0^N)\) is event detectable, and \(\omega_a\) contains no faulty transitions, then the marking \(M_k\) such that \(M_0 \xrightarrow{\alpha} M_k\), can be computed using the IPN state equation.

Since \(p_i^N\) has a token at \(M_k\), then a transition \(t_i^f \in (p_i^N)^* \cap T^f\) could be fired reaching a faulty marking \(M_t\). Thus, we will prove that it is possible to detect that \(M_t\) is reached.

Assume that such transition \(t_i^f\) at \(M_k\). Thus \(M_0 \xrightarrow{\alpha} M_k \xrightarrow{e} M_j\) and \(M_j\) adds a token into a place \(p_j^f \in (p_j^N)^* \cap P^f\) and removes a token from the place \(p_j^N\). Since the use of the modeling methodology assures that \(\lambda(t_i^f) = e\) and \(\phi(M_j) = \phi(M_j)\), then the firing of \(t_i^f\) cannot be detected using the current sequence of input-output symbols, i.e. it is not possible to determine when the faulty marking \(M_j\) was reached. In other words, the input-output symbol word \(\omega\) generates both \(\omega_a\) and \(\omega_o t_i^f.\) In order to detect when \(M_j\) is reached, we proceed as follows.

Since it holds that \(\forall t, \exists j, X_t(j) \geq 1, t_i \in T - T^f,\) then no T-semiflow can be fired without firing a transition in \((p_i^N)^* - T^f.\) Moreover, since the number of transitions in \((Q,M_0)\) is finite, then the length of a sequence attempting to fire a transition \((p_i^N)^*\) should be finite. Then a transition \(t_k \in (p_i^N)^* \cap T^N\) must be attempted to fire after the firing of a finite sequence.

Since the current marking in the net is the faulty marking \(M_j\) then \(t_k\) cannot be fired. Since the condition 2 indicates that \(\lambda(t_i) \neq e,\) then we can detect when the symbol \(\lambda(t_i)\) is given to...
the system. Moreover, since \( \bullet(t_k) = \{p^N_i\} \), if \( t_k \) cannot be fired, then certainly that \( M\left(p^N_i\right) = 0 \) and that \( M\left(p^N_j\right) = 1 \), where \( p^N_i \in (p^N_i)^T \cap P^F \). Thus the faulty marking \( M_j \) is detected, i.e. the IPN is input-output diagnosable. □

3.1 Diagnosability test
Determining when an IPN is input-output diagnosable is reduced to the following tests.
1. Binarity of \((Q,M_0)\). It is fulfilled because of the modeling methodology.
2. Liveness of modules can be tested efficiently; however the property is not preserved during arbitrary module composition operators. Thus some constraints in the application of synchronous and permissive compositions are introduced to guarantee liveness during composition. Such constraints may follow the rules introduced in (Koh & DiCesare, 1991) for module composition.
3. Event detectability and strongly connectedness are determined in polynomial time as well as detecting places \( p^N_i \in PN \) such that \((p^N_i)^T \cap T^F \neq \emptyset\). Then condition 2 of previous theorem is efficiently tested.
4. Finally, condition 1 can be verified in polynomial time. In this case we need to check that there exists no T-semiflow that does not include transitions in \((p^N_i)^T\), \( p^N_i \in PN \) such that \((p^N_i)^T \cap T^F \neq \emptyset\). Thus we need to check that the following linear programming problems have no solutions.

\[
\forall p^N_i \in PN \text{ such that } p^N_i \in PN, (p^N_i)^T \cap T^F \neq \emptyset
\]

\[
\exists X \text{ s.t. } CX = 0, \quad x(j) = 0, \forall t_j \in (p^N_i)^T - T^F
\]

There exist different strategies for constructing the diagnoser-model, we presented two different ways: a centralized and a distributed diagnoser. A centralized diagnoser model is composed by a copy of the normal system behavior of \((Q,M_0)\). Also a reduced diagnoser model can be built. Also, this work handles a distributed diagnoser which is very useful for large and complex system besides incorporate reliability to the system diagnosis process through redundancy techniques applied to the distributed diagnoser model. The different diagnoser models are defined in the following sections.

3.2 Centralized diagnoser design
Diagnostability theorem given above provides the basis for designing an on-line diagnoser (see figure 5). Such diagnoser can detect a fault and locate faulty markings reached by an IPN. The proposed scheme for diagnosis (Ramírez-Treviño, et al., 2004) handles a copy of the normal behavior model which must evolve similarly to the system; the outputs of both the system and the model are compared and, when there is a difference, a procedure is started to compute the faulty marking.

Example 2. The IPN system model depicted in figure 3 represents the normal behavior model for the producer-consumer diagnoser. The initial marking \( M_0 \) represents that all the
buffers are empty, the PU is waiting to deliver and the CU state is idle. Since this IPN is input-output diagnosable by theorem 1, then we can detect and locate the fault with the diagnoser. The system and its on-line diagnoser are depicted in figure 6, notice the diagnoser is a copy of the system. Now assume that the events represented by the sequence $t_2t_3$ are executed into the system, then the sequence $t_2t_3$ is fired in the diagnoser model. Thus both, the system and the diagnoser-model have the same output "producing" and "consuming". If the fault transition $t_{11}$ is fired, then $p_{11}$ is marked in the system and no system output change is detected. When the symbol of $t_1$, $\lambda(t_1)$ is given as input to the system, then the diagnoser-model evolves and its new output is "consuming", however the output of the system continues in "producing" and "consuming". Then the difference is the signal "producing"; thus the algorithm determines that $p_{11}$ is marked, thus the fault is isolated.

Fig. 5. On-line diagnose scheme

Fig. 6. On-line diagnoser model
4. Reduced diagnoser

4.1 Diagnoser model

**Definition 2**: The proposed diagnoser model structure for the system normal behavior \((Q^N, M_0^N)\) is an IPN \((Q^d, M_0^d)\) where the set of places is \(P^d = \{p_d\}\) and the set of transitions is \(T^d = T^N\), the incidence matrix \(C^d\) of \((Q^d, M_0^d)\) is the following

\[
C^d = B^T \varphi^N C^N
\]  

(2)

where \(B^T\) is a \(q\times 1\) vector \((q\) is the number of measurable places of \(((Q^N, M_0^N))\)), \(\varphi^N\) is the output matrix of \((Q^N, M_0^N)\), \(C^N\) is the incidence matrix of \((Q^N, M_0^N)\).

The initial marking of the diagnoser model structure for one place is computed as:

\[
M_0^d = B^T \varphi^N (M_0^N) \]  

(3)

We propose a matrix \(B\) that is computed as follows:

**Algorithm 1: Building B**

**Inputs**: \(C\)-incidence matrix of an \(IPN\),  
1 - number of places in the diagnoser-model,  
q - number of measurable places in the \(IPN\),  
**Outputs**: A matrix \(B\)

1. The “base number” \(b\) should be computed. In this case \(b = 2\max(\text{abs}(c_{ij})) + 1\), where \(c_{ij}\) is an element of incidence matrix \(C\).
2. Define a \(q \times 1\) vector.
3. \[
\begin{bmatrix}
    b^0 \\
    b^1 \\
    \vdots \\
    b^{q-1}
\end{bmatrix}
\]

This procedure computes matrix \(B\).

According to the way in which \(B\) was constructed, all columns of \(C^d\) must different from zero and different from each other.

If a transition \(t_i \in T - (T^R \cup T^F)\) is fired in \((Q, M_0)\) then it is fired in \((Q^d, M_0^d)\) (it is possible since these transitions are event detectable, thus the output system information is enough to detect when one of these transition is fired).

If a transition \(t_j \in T^R\) is enabled in \((Q_0, M_0^d)\) and \(\lambda(t_j)\) is activated in \((Q,M_0)\), then \(t_j\) must be fired in \((Q^d, M_0^d)\). Thus, if \(t_j\) is not fired in \((Q,M_0)\), then \((Q,M_0)\) reached a faulty marking. In this case the output of the system and the output of the diagnoser are different from each other.

4.2 Error computation and fault isolation

**Definition 3**: Error computation. The k-th error is computed by the following equation:

\[
e_k = M_k^d - B^T(\varphi M_k)\]  

(5)

Notice that \(e_k\) is computed from the diagnoser-model output and not from the marking \(M_k\).

It means that the proposed diagnoser is using the system output and not internal system signals (those signals that are non measurable).

**Definition 4**: Fault isolation. When \(e_k \neq 0\), an error is detected, then a faulty marking was reached. The mechanism used to find out the faulty marking is named fault isolation. This work proposes the following algorithm to accomplish this task.
Algorithm 2: Fault isolation

Inputs: $M_k, M_k^d, e_k$
Outputs: $p$(faulty place), $M_f$(faulty marking)
Constants: $C^d$ is the IPN diagnoser structure incidence matrix

\( i = \text{index of the column of } C^d \text{ such that } C^d(1,i) = e_k \)
\(-\ \forall p \in \cdot t, M_k(p)=0 \)
\(-\ \forall p \in t \cdot, M_k(p)=0 \)
\(-\ \forall p \in (t_i)^* \cap P^F, M_k(p^F)=1 \)
\(-\ M_f = M_k \)
- Return $(p, M_f)$

Example 3. Consider the system of the example 1. The IPN depicted in figure 4 represents the behavior of the system of example 1. Since this IPN is input-output diagnosable by theorem 1, then a diagnoser can be built for this system. In this case we will use the reduced structure presented in this section.

The base obtained to compute $B$ is $b=2*1+1=3$; since we build $B$ using algorithm 9. We obtain the following vector:

$$B^r = \begin{bmatrix} 1 & 3 & 9 & 27 \end{bmatrix}$$

Therefore $C^d$ is:

$$C^d = \begin{bmatrix} -1 & 27 & 1 & 9 & -9 & -27 & 3 & -3 \end{bmatrix}$$

Hence, its associated IPN is depicted in figure 7.
The initial marking of the diagnoser is $M_0d = [3]$. In order to show how the diagnoser works, assume that the following sequence $t_2t_3$ is executed into the system, then this sequence is fired in the diagnoser. Thus the system output is $(\varphi(M_s))^T = [1 \ 1 \ 0 \ 1]$, and the marking of the IPN diagnoser is $M_{1d} = [3 \ 1]$. Then $e_k = M_{0d} - (B^T)(\varphi M_k) = [31] - [31] = 0$, thus the system is in a normal state. Now if the faulty transition $t_9$ is fired, then $p_{11}$ is marked, however no change in the output system is detected. If the symbol of $t_1$ ($\lambda(t_1) = a$) is given as input to the IPN of the system model and IPN diagnoser, then the diagnoser evolves, and $M_{k+1} = [30]$. Then $e_k = -1$ indicating the existence of an error. The fault isolation algorithm (algorithm 2) detects that the column 1 of $C_4$ is equal to $e_9$, thus $t_1$ was not fired in the system. Then the same algorithm detects the faulty marking and determines that the faulty place $p_{11}$ is marked.

5. Distributed diagnoser

5.1 Model distribution

In order to build a distributed diagnoser, the IPN model $(Q, M_0)$ can be conveniently decomposed into $m$ interacting modules where different modules share nodes (transitions and/or places).

**Definition 5.** Let $(Q, M_0)$ be an IPN. A module $\mu_k = (N_k, \Sigma_k, \lambda_k, \phi_k)$ is an IPN subnet of the global model $(Q, M_0)$, where:

- $N_k = (T_k, P_k, I_k, O_k, I^C_k, O^C_k, M_{0k})$ where:
  - $T_k \subseteq T$,
  - $P_k = P_k^l \cup P_k^C$; $P_k^l \subseteq P$; $P_k^C$ represents the communication places among modules; this set is a copy of some places $P_k^l$ that belongs to other modules, $l \neq k$. $P_k^C$ is the minimal places that is required for the transitions of the module are event-detectable. $M(P_k^C) = M(P_k^l)$.
  - $I_k(O_k): P_k^l \times T_k \to Z^*$, s.t., $I_k(p, t) = I(p, t)$, $O_k(p, t) = O(p, t)$, $\forall p \in P_k^l$ and $\forall t \in T_k$.
  - $I_k^C(O_k^C): P_k^C \times T_k \to Z^*$, s.t., $I_k^C(p, t) = I(p, t)$, $O_k^C(p, t) = O(p, t)$, $\forall p \in P_k^C$ and $\forall t \in T_k$.

- $\lambda_k: T_k \to \Sigma \cup \{\varepsilon\}$, s.t., $\lambda_k(t) = \lambda(t)$ and $t \in T_k$.
- $\phi_k: R(\mu_k, M_{0k}) \to (Z^*)^\mu$, $\phi_k$ is restricted to the outputs associated to $P_k$.

**Definition 6.** Let $(Q, M_0)$ be an IPN. A distribution $D_{Ni}$ of $(Q, M_0)$ is a finite set of $m$ modules, i.e., $D_N = \{\mu_1, \mu_2, \ldots, \mu_m\}$. The distribution $D_{N_i}$ holds the following conditions:

1. $T^e_i = T^c_i \cup T^l_i$ (It is a partition over $P^C_i$).
2. $T^e_i = T$ (It is not a partition)

The set of communication places $P_{com} = \bigcup_{i=1}^{n} P^C_i$ of a distribution represents the measurable places of each module $\mu_k \in D_{Ni}$, needed to guarantee the event-detectability property of $\mu_k$.

**Definition 7.** Let $D_{Ni}$ be a distribution of $(Q, M_0)$ and $\sigma = t_1 \ldots t_n ...$ be a sequence of $E(Q, M_0)$. We define the natural projection $\Pi_k$ of $E(Q, M_0)$ over the languages of the modules $\mu_k \in D_{Ni}$ of the following way:

$$\Pi_k: E(Q, M_0) \to E(\mu_k, M_{0k})$$

$$\forall t_1, t_2, \ldots, t_n \in E(Q, M_0)$$
\[ PT_k(\varepsilon) = \varepsilon \]
\[ PT_k(t_1 \ldots t_q) = \begin{cases}  
PT_k(t,t_1 \ldots t_q) & \text{if } t \notin T_k \\
PT_k(t_1 \ldots t_q) & \text{if } t \in T_k 
\end{cases} \]

There exists the input-output symbol projection over the input and output module symbols. Let \( \omega = (\alpha_0, y_0, \alpha_1, y_1, \ldots, \alpha_n, y_n) \) be a sequence of symbols of \((Q,M_0)\)

\[ P\Lambda_{\mu_k}(\omega) = (P_{\text{in}}\alpha_0, P_{\text{out}}\gamma_0, P_{\text{in}}\alpha_1, P_{\text{out}}\gamma_1, \ldots, P_{\text{in}}\alpha_n, P_{\text{out}}\gamma_n) \]

where:

\[ P_{\text{in}}\alpha_i = \begin{cases}  
\varepsilon & \text{if } \alpha_i \notin \sum_k \\
\alpha_i & \text{if } \alpha_i \in \sum_k 
\end{cases} \]

\[ y'_i(s) = 0 \text{ if the measurable place does not belong to } \mu_k \]

\[ y'_i(s) = y_i(s) \text{ otherwise.} \]

**Example 4.** Consider the IPN system model depicted in the figure 4. We partition the producer-consumer model to obtain a distributed model. The figure 8 depicts a distribution \(DN_i\). For the sake of simplicity, we use in the example the same names for duplicated nodes (places or transitions) belonging to different modules. The distribution has three modules, i.e., \(|DN_i| = 3\); \(L^C(Q_i^C)\) is represented by the dashed arcs. The module \(\mu_1\) has the transitions \(T_1 = \{t_1,t_2,t_3,t_9\}\) and the place set \(P_1 = \{p_1, p_2, p_3, p_{11}\}\cup\{p_{10}\}\). We are preserving the property of event detectability of common transitions duplicating measurable places, and using these copies in different modules.

![Fig. 8. Distributed Interpreted Petri Net Models](image-url)

**5.2 Distributed input-output diagnosability**

The results of centralized diagnosability are applied to the modules. A module is locally diagnosable if, for every local fault we can detect it only using local information, else it is conditionally diagnosable.

**Definition 8.** (Local Diagnosability) A module \(\mu_k \in DN_i\) is said to be locally input-output diagnosable in \(k < \infty\) steps if any faulty marking \(M_{\text{fin}} \in R(\mu_k, M_{\text{ini}})\) is distinguishable from any other \(M_{\text{fin}} \in R(\mu_k, M_{\text{ini}})\) using words \(\omega_n\), where \(P\Lambda_{\mu_k}(\omega) = \omega_n\).
Definition 9. (Conditional Diagnosability) A module $\mu_n \in DN_i$ is said to be conditional input-output diagnosable in $k < \infty$ steps if any faulty marking $M_{fn} \in R(\mu_n, M_{0n})$ is distinguishable from any other $M_{kn} \in R(\mu_n, M_{0n})$ using words $\omega_n$ and $\omega_m$, where $P\Lambda_{\mu} (\omega) = \omega_n$ and $P\Lambda_{\mu} (\omega) = \omega_m$, where $\omega_m$ denotes the set of all input-output sequences that lead to a marking a duplicate place in $\mu_m$, where $\mu_n \neq \mu_m$.

5.3 Communication channels

Definition 10. The border transitions between two modules $\mu_k$ and $\mu_i$: $T_{k\text{border}}^i = \{ t_i \mid t_i \in \{ T_k \cap T_i \} \}$. This concept can be extended to several modules.

The communication channels between two modules are represented by $I_i(\partial_k C_i)$. We assume that every module can communicate with every other module. The firing of a transition $t_i \in T_k$ may be local to the module $\mu_k$ and cause only a local marking change, or it may involve communication with another module if $t_i \in T_{k\text{border}}^i$. Every time a transition $t_i \in T_{k\text{border}}^i$ is fired in $\mu_k$, then a message is sent to every $\mu_i$ containing the same $t_i \in T_{k\text{border}}^i$ to put a token in some place $p_x \in P_i C_i$, such that $p_x \in t_i$. In (Lampson, 1993) it is proposed different ways to create protocols for implementing reliable messages.

5.4 Redundant diagnoser

Since distributed diagnosers lead to the use of several computers (CPU), then, redundancy can be introduced in the diagnosers. For instance, the Triple Modular Redundancy (TMR) can be used in this case.

Assume that a distribution $DN_i=\{\mu_1, \mu_2, \ldots, \mu_m\}$ was obtained and that it was distributed over $m$ computers (see figure 9).

Fig. 9. On-Line Distributed Diagnoser

Then, a TRM scheme can be also applied to figure 9 (see figure 10), increasing the diagnoser reliability.
6. Conclusions

This chapter introduced the diagnosability property in DES modeled using IPN. It presented a structural characterization of this property using the T-semiflows of the IPN. The approach herein presented exploits the IPN structure to determine when it is diagnosable, this approach leads to polynomial characterization of diagnosability. Based on the DES model, three different types of diagnosers were presented. The first one was a centralized version, allowing to detect and locate faults. Sometimes, however, the system could very large; leading to large diagnoser models thus the other two diagnosers are designed to tackle this problem.

The second diagnoser is a reduced scheme. It uses one place; however the number of tokens could be large. The third diagnoser is a distributed one, were the diagnoser model is distributed over different computers. Adopting this approach the problems that appear in centralized versions are eliminated. Moreover, in this last case, a redundant diagnoser can be used for increasing the reliability of the distributed diagnosers.

7. References


The book presents an excellent overview of the recent developments in the different areas of Robotics, Automation and Control. Through its 24 chapters, this book presents topics related to control and robot design; it also introduces new mathematical tools and techniques devoted to improve the system modeling and control. An important point is the use of rational agents and heuristic techniques to cope with the computational complexity required for controlling complex systems. Through this book, we also find navigation and vision algorithms, automatic handwritten comprehension and speech recognition systems that will be included in the next generation of productive systems developed by man.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following: