Congr. Int. en Ing. Electrónica. Mem. ELECTRO, Vol. 40, pp. 126 - 132, Oct 2018, Chihuahua, Chih. México
http://electro.itchihuahua.edu.mx/memorias_electro/MemoriaElectro2018.zip
ISSN 1405-2172

# ANALYSIS AND IMPLEMENTATION OF AN ALGORITHM TO DETECT FAULTS

*Hernández-Rueda[1] K., Meda-Campaña[2] M.E.*
Universidad de Guadalajara
[1]CUCSUR-Ingenierías, [2]CUCEA-Sistemas de Información
Av Juárez 976, Col. Centro.
52-01-317-3825010 ext 57080.
[1]karen.hrueda@academicos.udg.mx, [2]emeda@cucea.udg.mx.

## RESUMEN

**El artículo presenta una corrección necesaria sobre un algoritmo propuesto en la literatura para determinar si un sistema de eventos discretos modelado en redes de Petri Interpretadas es diagnosticable con el objetivo de detectar la ocurrencia de faltas. Lo relevante de este trabajo es que expone el análisis hecho sobre el algoritmo y se presenta su implementación en Maple con la corrección propuesta.**

***Palabras clave: Sistema de eventos discretos, Diagnóstico de faltas, redes de Petri interpretadas, Maple.***

## ABSTRACT

**The aim of this paper is to present a modification of a proposed algorithm to determine if a discrete event system described by an Interpreted Petri net is diagnosable allowing to detect the occurrence of faults by an external agent. The relevancy of this work is to state the required condition to modify the algorithm, present the implementation of the modified algorithm in Maple and analyze its functionality.**

***Keywords: Discrete Event System, Fault diagnosis, Interpreted Petri nets, Maple.***

## 1. INTRODUCTION

Although the systems are designed and implemented to exhibit specific performance (characteristics defined by the subsystems and their interrelationship) where faults are a priori known (unwanted behavior). The occurrence of these faults is practically impossible to predict, in the first place because they occur asynchronously and in second place they can be fired by system exogenous events, leading to unspecified system behaviors changing dramatically the system performance.

In general, since all systems are subject to fail in some way, it is desirable to have a mechanism or procedure to automatically determine when and where a fault has occurred. When the faults of a system can be automatically detected, then it is said that the system exhibits the diagnosability property or that the system is diagnosable. The process used to detect and locate faults in a diagnosable system is named the fault diagnosis process.

Fault diagnosis of Discrete Event System (DES) is a research area that has received a great attention in the last years and has been motivated by the need of ensuring the correct and safe operation of large complex systems. The problem of fault diagnosis for DES using model-based diagnosis have been successfully used in a wide class of technological systems, ranging from document processing systems to intelligent transportation systems [1]. There exists a lot of literature on model-based diagnosis; several representative works use finite automaton (FA) or Petri nets (PN) as modeling formalism. Although automaton models are suitable for describing DES, the use of PN offers significant advantages because of their twofold representations: graphical and mathematical [2]. Moreover, the intrinsically distributed nature of PN where the notion of state (marking) and action (transition) is local reduces the computationally complexity involved in solving a diagnosis problem. Therefore, PN are considered a suitable formal tool to carry out the study of fault diagnosis in DES. For example, Ushio et al. in [3] addressed the diagnosis approach proposed in [4] and uses the formalism of PN. They represent the normal and failure behavior of a system with PN. This contribution gave rise to the study of diagnosis based on models with PN. Basile and Tommasi in [5] presented sufficient conditions for diagnosability of DES modeled as PN. It is referred to the concept of diagnosability given by Sampath et al. [4]. Then, in [6] the design of an on-line diagnosis is presented, it defines and solves some problems of integer linear programming (ILP). Dotoli et al. in [7] proposed a diagnoser that works on-line in order to avoid the redesign and the redefinition of the diagnoser when the structure of the system changes. The proposed approach is a general technique since no assumption is imposed on the reachable state set that can be unbounded, and only few properties must be fulfilled by the structure of the PN modeling the system fault behavior. In [8] it is assumed that there exist not unobservable cycles no blocked firing sequences after the firing of any faulty transition; necessary and sufficient conditions are given for diagnosability based in reachability diagnoser. In [9] is proposed a structural characterization for faults diagnosis and a reduced diagnoser that ensures the fault detection and location in a finite number of steps. They used Interpreted PN (IPN) to model the system and its faults. In [10] a new structural diagnosability characterization for each of permanent and operational faults considering some relationship between conservative parts, repetitive parts and siphons are proposed. These studies have several advantages over previous work because they perform the structural analysis of PN models instead of reachability analysis to determine the diagnosability of a DES. In addition, they propose on-line diagnosers scheme that are easy to implement. Therefore, in this work is analyzed and implemented the algorithm (written in Maple) to determine

when an IPN exhibits the diagnosability property. It is based mainly on the characterization of diagnosability property in IPN proposed by [10]. The contributions of this work are; 1) the condition required to properly applying the algorithm to determine if the IPN is diagnosable and 2) implementation of algorithms to detect when an IPN is diagnosable.

The paper is organized as follows: section 2 provides basic definitions of DES, PN, and IPN. Section 3 presents the theory of fault diagnosis problem. In the section 4 is defined and analyzed the algorithm to determine the diagnosability of an IPN and besides that its implementation in Maple is shown. Finally, in section 5, the conclusion and future work are presented.

## 2. BACKGROUND

### 2.1. Discrete Even System

A DES has the state set naturally described by a discrete set like {0, 1, 2,...} and transitions are observed at discrete points in time. These state transitions are associated with events. Examples of these systems are network systems, distributed systems, traffic control systems, manufacturing systems, among others [11].

### 2.2. Petri nets

A PN is represented by two kinds of vertices (figure 1): circles, represent places ($p_1$,$p_2$,$p_3$), that are associated with actions or system outputs to be modeled and bars o rectangles ($t_1$,$t_2$,$t_3$,$t_4$) representing transitions, which are associated with events and actions or outputs. The input places (output places) are places whose arcs lead to (leave to) a transition $t_j$ and they are considered input (output) of $t_j$. An initial marking ($M(p_1)$ =1) would be an initial distribution of marks (black dots into the places). The presence or absence of a mark in a place can indicate whether a condition associated with this place is true or false. At any given time, for instance, the distribution of marks into places is called PN marking. The marking defines the current state of the modeled system.

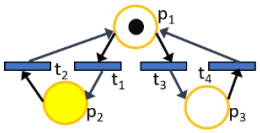


*Fig. 1. Petri net*

The PN works in order to simulate the dynamic behavior of a system, the marking in a PN is changed according to the following firing rule: a) a transition "t" is said to be enabled if each input place "p" of "t" is marked with at least w(p,t) tokens, where w(p,t) is the weight of the arc from "p" to "t", b) an enabled transition may or may not fire (depending on whether or not the event related with "t" actually takes place), and c) the firing of an enabled transition "t" removes w(p,t) tokens from each input place "p" of "t", and adds w(t,p) tokens to each output place "p" of "t", where w(p,t) is the weight of the arc from "t" to "p". The formal definition of PN is presented as follows and it is taken from [12].

*Definition 1*: A Petri Net structure G is a bipartite digraph represented by the 4-tuple G=(P,T,I,O) where:

- $P = \{p_1, p_2, ..., p_n\}$ and $T = \{t_1, t_2, ..., t_m\}$ are finite sets of vertices called places and transitions, respectively.
- $I(O) : P \times T \rightarrow Z^+$ is a function representing the weighted arcs going from places to transitions (transitions to places); $Z^+$ is the set of nonnegative integers.

The symbol $\bullet t_j$ denotes the set of all places $p_i$ such that $I(p_i, t_j) \neq 0$ and $t_j\bullet$ the set of all places $p_i$ such that $O(p_i, t_j) \neq 0$. Analogously, $\bullet p_i$ denotes the set of all transitions $t_j$ such that $O(p_i, t_j) \neq 0$ and $p_i\bullet$ the set of all transitions $t_j$ such that $I(p_i, t_j) \neq 0$.

The pre-incidence matrix of G is $C^- = [c_{ij}^-]$, where $c_{ij}^- = I(p_i, t_j)$; the post-incidence matrix of G is $C^+ = [c_{ij}^+]$, where $c_{ij}^+ = O(p_i, t_j)$; the incidence matrix of G is $C = C^+ - C^-$. The marking function $M: P^- \rightarrow Z^+$ represents the number of marks (depicted as dots) residing inside each place. The marking of a PN is usually expressed as an n−entry vector. M function can be represented as M(p).

*Definition 2*: A PN is the pair $N = (G, M_0)$, where G is a PN structure and $M_0$ is an initial token (mark) distribution over places.

*Definition 3*: A P−semiflow $Y_i$ (T− semiflow $X_i$) of a PN is a positive integer solution of the equation $Y_i^T C = 0$ ($CX_i = 0$). The support of the P−semiflow $Y_i$ (T− semiflow $X_i$) is the set $\|Y_i\| = \{p_j \mid Y_i(p_j) \neq 0\}$ ($\|X_i\| = \{t_j \mid X_i(t_j) \neq 0\}$).

*Definition 4:* The reachability set of N, denoted by $R(N, M_0)$, is the set of all possible reachable markings from $M_0$, firing only enabled transitions.

*Definition 5*: A PN $(N, M_0)$ is *k*-safe (*k*-bounded) if for all $M \in R(N, M_0)$ and places $p \in P$, $M(p) \leq k$. 1 − safe nets are simply called safe.

*Definition 6*: A PN $(N, M_0)$ is live if for all $M_i \in R(N, M_0)$ and for all $t \in T$ it is true that $\exists M_j$, such that $M_i \rightarrow M_j \rightarrow$ .

*Definition 7*: A siphon is a subset of places $S = \{p_1, ..., p_s\} \subseteq P$ of a PN such that the set of input transitions $\bullet S$ is contained in the set of output transitions $S\bullet$, i.e., $\bullet S \subset S \bullet$.

### 2.3. Interpreted Petri net.

An IPN is an extension of PN based on the input and output signals. IPN is a PN that include input and output alphabets that are associated with the transitions and places, of PN, respectively. An IPN can model the commands sequences given by the signals from actuators and sensors each time a new state is reached. Graphically, an IPN is presented in figure 2. The measurable places $\{p_1, p_3\}$ of IPN are

transparent circles while the no-measurable place $\{p_2\}$ is a dark circle. The formal definition is taken from [12] and it is as follows:

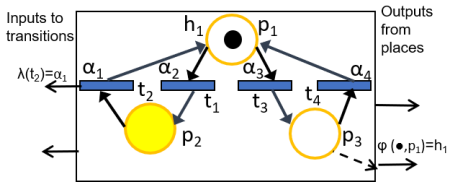


*Fig. 2. Interpreted Petri net*

*Definition 8*: An IPN is the 4-tuple Q = (G, Σ, λ, φ) where:

- N = (G, $M_0$) is a PN.
- Σ = {$\alpha_1$, $\alpha_2$, ..., $\alpha_r$} is the input alphabet of the net, where $\alpha_i$ is an input symbol.
- λ : T → Σ ∪ {ε} is a labeling function of transitions with the following constraint: ∀$t_j$, $t_k$ ∈ T, j ≠ k, if ∀$p_i$ I($p_i$, $t_j$) = I($p_i$, $t_k$) ≠ 0 and both λ($t_j$) ≠ ε, λ($t_k$) ≠ε, then λ($t_j$) ≠ λ($t_k$). In this case ε represents an uncontrollable system event.
- There exists a q×n matrix φ, such that $y_k$ = φ$M_k$ is mapping of the marking $M_k$ into the q−dimensional observation vector. Column φ(•, i) is the elementary vector $e_h$ if place $p_i$ has associated the sensor place h; or the null vector if $p_i$ has no associated sensor place. In this case, an elementary vector $e_h$ is the q−dimensional vector with all its entries equal to zero, except entry h, that it is equal to 1. A null vector has all its entries equal to zero.

Notice that q places have associated a sensor, signal thus they are measurable or observable.

A transition $t_i$ ϵ T of an IPN is enabled at marking $M_k$ if ∀$p_i$ ∈ P, $M_k(p_i)$ > I($p_i$, $t_i$). An enabled transition $t_j$, labeled with a symbol other than ε (empty or silent) symbol, must be fired when λ($t_j$) is activated. An enabled transition $t_j$, labelled with a ε symbol can be fired. When an enabled transition $t_i$ is fired in a marking $M_k$, then a new marking $M_{k+1}$ is reached. This fact is represented as $M_k \xrightarrow{t_j} M_{k+1}$; $M_{k+1}$ can be computed using the dynamic part of the state equation represented by (1):

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

*Definition 9*: A firing transition sequence of an IPN (Q, $M_0$) is a sequence σ = $t_i t_j$ ... such that $M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j}$ ... The set of all firing sequence £(Q, $M_0$), is called the firing language of (Q,$M_0$). £(Q, $M_0$) ={ σ | σ = $t_i t_j$ ... where $M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j}$ ... }.

*Definition 10*: A sequence of observation vectors (output symbols) of (Q, $M_0$) is a sequence ω= ($y_0$) ($y_1$) ... ($y_n$), where $y_k = \phi M_k$ and $y_i \neq y_{i+1}$. If ω is a sequence of output symbols, then the set of firing transition sequences σ ∈ £(Q, $M_0$) whose firing generates the output sequence ω is represented by Ω(ω).

*Definition 11*: Let (Q, $M_0$) be an IPN. The set Λ(Q, $M_0$) denotes all sequences of output symbols of (Q,$M_0$). The set of all output sequences of length greater than or equal to k will be denoted by $\Lambda^k$ (Q, $M_0$), i.e., $\Lambda^k$ (Q, $M_0$) = {ω ∈ Λ(Q, $M_0$) | |ω|≥ k}.

*Definition 12*: The set of all output sequences leading to an ending marking in the IPN (Q, $M_0$) is denoted by $\Lambda_B$(Q, $M_0$), i.e., $\Lambda_B$ (Q, $M_0$) = {ω ∈ Λ(Q, $M_0$) | ∃σ ∈Ω(ω) such that $M_0 \xrightarrow{\omega} M_j$ and $M_j$ enables no transition, or when $M_j$ enables $t_i$ ($M_0 \xrightarrow{t_i}$) then C(●, $t_i$) = $\vec{0}$}.

## 3. DIAGNOSIS PROBLEM

### 3.1. Diagnosis based on model

This model represents the diagnosis system (figure 3) and has a diagnoser (a monitoring online system whose warns the presence of faults) that compares the normal behavior ($Q^N$,$M_0^N$) without faults modeled with IPN, with the current behavior (Q, $M_0$) of a DES, when there exists a difference between these behaviors, then a fault is detected and it can be seen as an error ($e_k$) in the DES [12].

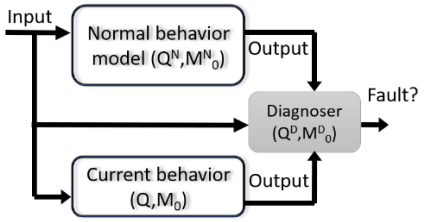


*Fig. 3. Online diagnosis based on model*

### 3.2. Event-detectability property

If the input-output sequences can be detected using only output signals (systems sensor outputs) and the structural information of the IPN, this net will be called event-detectable.

*Definition 13*: An IPN (Q, $M_0$) is event-detectable if the firing of any transition can be uniquely determined from knowledge of the input and output produced by (Q, $M_0$) [13].

*Definition 14*: An IPN (Q, $M_0$) is event-detectable iff ∀σ ∈ £(Q, $M_0$), the firing of any pair of transition $t_i$,$t_j$ ∈ σ, can be distinguished from each other using the information in ω ∈ Λ (Q, $M_0$) [13].

The following lemma gives a polynomial characterization of event-detectable IPN that is used to determine if an IPN is diagnosable [6]. This means that if an IPN is not event-detectable it is impossible to know if it can be diagnosable.

*Lemma 1*: A live IPN given by (Q, $M_0$) is event detectable iff

- ∀$t_i$, $t_j$ ∈ T such that λ($t_i$) = λ($t_j$) or λ($t_i$) = ε it holds that φC(•, $t_i$) ≠ φC(•, $t_j$), and
- ∀$t_k$ ∈ T it holds that φC(•, $t_i$) ≠0.

### 3.3. Modeling of faults

The faults are represented by transitions in the IPN, i.e., it is considered that a fault event modeled occurs when a transition that is considered a fault is fired according to the rules and properties of the PN. The objective is to diagnose the occurrence of fault events based on the sequence of observed events. Furthermore, the faults that are considered here are permanent ones. A permanent fault occurs when a task stops its execution while other(s) task(s) can be continued to run in the system.

This work deals with systems whose normal behavior model ($Q^N$, $M_0{}^N$) can be represented by a live and safe IPN. Once the DES is described by a live and safe IPN, the next step is to represent faults into the normal behavior model. This is done based on [12]. When a permanent fault occurs, then one task is stopped while other concurrent tasks may continue their execution. In permanent faults, the involved faulty devices will remain in a faulty state until they are repaired. The proposed modeling strategy for representing faults is straightforward. Consider first the model ($Q^N$, $M_0{}^N$) that describes the normal functioning of the system. Then for every place $p_i^N$ representing an operation at which a fault may occur, add an uncontrollable transition $t_f$ , a faulty place $p_i^F$, and the arcs ($p_i^N$, $t_f$ ) and ($t_f$ , $p_i^F$). The new faulty place $p_i^F$ must be labeled with the same symbol that $p_i^N$ for stating that the fault cannot be detected from the observation of the outputs. The obtained model describes both normal and faulty behavior as it can see in figure 4. The permanent faults are $t^1_{pf}$ and $t^2_{pf}$ [12].

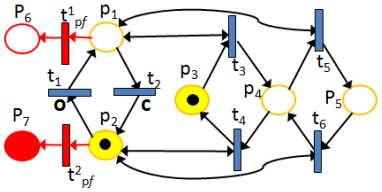


*Fig. 4. Normal and faulty behavior of an IPN*

### 3.4. Diagnosability

The diagnisability problem consist in determining if a system is diagnosable, i.e., if the occurrence of a fault can be detected in a finite number of steps, using the input-output system information. Following notation and definitions will be useful in the diagnosability analysis.

The set of places P of an IPN (Q, $M_0$) is partitioned into two subsets, $P = P^F \cup P^N$ where $P^F$ is the set of places coding faulty states, and $P^N$ is the set of places coding normal states of the IPN. The markings in R(Q, $M_0$) can also be partitioned into the following two subsets: $F = \{M \in R(Q, M_0) \mid \exists p_k \in P^F$ such that $M(p_k) > 0, M \in R(Q, M_0)\}$ and $R(Q^N, M_0{}^N) = R(Q, M_0) - F$, where F is the set of the faulty markings and $R(Q^N, M_0{}^N)$ is the set of the normal states. The embedded normal behavior IPN ($Q^N$, $M_0{}^N$) of (Q, $M_0$) is the IPN included in (Q, $M_0$) when $P^F$ and $T^F = \bullet P^F$ are not considered. In ($Q^N$, $M_0{}^N$) the set of places is $P^N = P - P^F$, the set of transitions is $T^N = T - T^F$ and the set of arcs of ($Q^N$, $M_0{}^N$) is $A^N = ((P^N \times T^N) \cup (T^N \times P^N)) \cap (A)$, where $A = \{(p_i, t_j) | p_i \in P, t_j \in T$ and $I(p_i, t_j) = 1\} \cup \{(t_i, p_j) | p_i \in P, t_j \in T$ and $O(p_i, t_j) = 1\}$.

*Definition 16*: Let (Q, $M_0$) be an IPN, $P^N$ be the normal set of places, and $T^F$ be the set of faulty transitions of (Q, $M_0$). The set of risky places of (Q, $M_0$) is $P^R = \bullet T^F$. The post-risk transition set of (Q, $M_0$) is $T^R = \{P^R\bullet \cap T^N\}$[13].

Considering last definition, the next proposition [10] is used to determine if a permanent fault can be diagnosable.

*Proposition 1*: Let (Q, $M_0$) be a safe ($Q^N$, $M_0{}^N$) that is safe, live and strongly-connected. Let $t_i$ be a permanent fault, $p_k$ be a risky place and $S_{ti}$ be the siphon that will be unmarked when $t_i$ is fired. Assume that $|p_k\bullet| = 1$ and the post-risky transition $t_a \in p_k\bullet$ and the pre-risky transitions are event detectable. (Q, $M_0$) is diagnosable with respect to $t_i$ if all the T-semiflows of the net contains transitions in $\bullet S_{ti} \cup S_{ti}\bullet$.

## 4. ALGORITHM TO DETECT FAULTS

### 4.1. Definition

Next algorithm is used to diagnose the permanent faults in an IPN and is taken from [10]. The inputs are the IPN (Q, $M_0$) and ($Q^N$, $M_0{}^N$) that are live and binary, and the output is the T-semiflow X that can be fired infinitely often when the permanent fault $t_j$ occurs.

*Algorithm 1* Permanent faults diagnosability of an IPN
Let $p_k$ be the risky place associated to the permanent fault $t_i$, $S_{ti}=\{p_k\}$. For each place $p_i \neq p_k$ solve the following Linear Programming Problems LPPs:

$$LPP_1 = \left\{ \begin{array}{c} Min \sum_{i=1}^{|P|} Y_1(i) \\ s.t. \\ Y_1^T C = 0 \\ Y_1(p_k) = 1 \\ Y_1(p_i) = 1 \\ 0 \leq Y_1(p_r) \leq 1 \end{array} \right\} \quad (2)$$

Where $p_i$ is the current place that is analyzed and $p_r$ is the rest of places. If (2) has no solution then places $p_i$, $p_k$ are in different minimal P-semiflow, go to label #. If (2) has a solution then places $p_i$,$p_k$ are in the same minimal P-semiflow or they are in two different minimal P-semiflows not sharing any place. $LPP_2$ will determine which the case is.

$$LPP_2 = \left\{ \begin{array}{c} Min \sum_{i=1}^{|P|} Y_2(i) \\ s.t. \\ Y_2^T C = 0 \\ Y_2(p_k) = 0 \\ Y_2(p_i) = 1 \\ 0 \leq Y_2(p_r) \leq 1 \\ Y_2(i) \leq Y_1(i) \end{array} \right\} \quad (3)$$

If (3) has a solution then places $p_i$, $p_k$ are in different minimal P-semiflows, go to label #. If (3) has no solution then places

$p_i$,$p_k$ are in the same minimal P-semiflow. $S_{ti}$= $S_{ti} \cup \{p_i\}$ because are in the same minimal P-semiflows.

Label # : end for.

Thus, the transition set $T^{ti}$= $\bullet S_{ti} \cup S_{ti} \bullet$ contains the transitions that will be immediately death if the faulty transition $t_i$ is fired. Now, $LPP_3$ computes if there exists a T-semiflow not containing transitions in $T^{ti}$. Let Z(h)=1 if $t_h \in T^{ti}$ and Z(h)=0 in otherwise.

$$LPP_3 = \begin{Bmatrix} Max \ \sum_{i=1}^{|T|} X(i) \\ s.t. \\ CX = 0 \\ Y_2(p_k) = 0 \\ X'Z = 0 \\ 0 \leq X(h) \leq 1 \end{Bmatrix} \quad (4)$$

Where $X'$ is the transpose of X. According with (4), if X is empty then this faulty transition $t_i$ will be diagnosable. In the algorithm, $S_{ti}$ has the places forming the siphon that will be unmarked when the faulty transition $t_i$ is fired. For sure that $p_k$ belongs to the siphon. Thus, $LPP_1$ computes if a place $p_i$ belongs to the same minimal P-semiflow than $p_k$ or if $p_i$ belongs to a minimal P-semiflow not sharing any place with the P-semiflow containing $p_k$. If $LPP_1$ has a solution, then $LPP_2$ determines if $p_i$ and $p_k$ belong to the same P-semmiflow: If it is the case then $S_{ti}$= $S_{ti} \cup \{p_i\}$. When all the places are analyzed, then $S_{ti}$ contains all the places belonging to the siphon that will be unmarked when $t_i$ is fired. Afterwards, $LPP_3$ computes a minimal T-semiflow that can be fired infinitely often even when $t_i$ is fired. Thus, if such T-semiflow exists, then $t_i$ is not diagnosable.

### 4.2. Analysis

To analyze how the previous algorithm works, consider the IPN depicted in figure 5 that is live, binary and strongly-connected. This contains a permanent fault $t^1_{pf}$=$t_6$ and the risky-place $p_k$=$p_2$, $t_i$=$t_6$ and $S_{ti}$=$p_2$. It can see that this IPN does not possess an indeterminate cycle, so this net must be diagnosable.

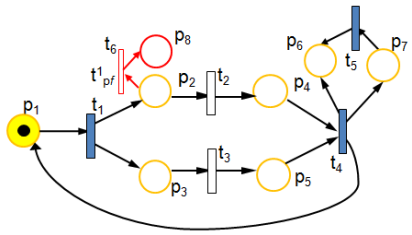


*Fig. 5. IPN live and safe with a permanent fault*

Applying the algorithm above presented, it is executed according the number of places of the IPN (for each place $p_i \neq p_k$).

Then, in cycle **For** $LPP_1$ the objective function is:

$$Min \ \sum_{i=1}^{|7|} Y_1(i) = Min\ (y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_7) \ \ where$$

$$C = \begin{vmatrix} -1 & 0 & 0 & 1 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 \end{vmatrix}, Y_1 = \begin{vmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{vmatrix}, \begin{matrix} -y_1 + y_2 + y_3 = 0 \\ -y_2 + y_4 = 0 \\ -y_3 + y_5 = 0 \\ y_1 - y_4 - y_5 - y_6 + y_7 = 0 \\ y_6 - y_7 = 0 \end{matrix} \quad (5)$$

$$For\ p_i \neq p_k, p_i = p_1\ and\ p_k = p_2, Y_1(y_2) = 1\ and\ Y_1(y_1) = 1.$$
$$The\ condition\ 0 \leq Y_1(p_r) \leq 1,\ implies\ that\ \ 0 \leq Y_1(y_3) \leq 1,$$
$$0 \leq Y_1(y_4) \leq 1, 0 \leq Y_1(y_5) \leq 1,\ \ 0 \leq Y_1(y_6) \leq 1, 0 \leq Y_1(y_7) \leq 1.$$
$$If\ y_1 = y_2 = 1 \rightarrow y_3 = 0, y_4 = 1,\ y_5 = 0,\ if\ y_6 = 0 \rightarrow\ y_7 = 0$$
$$then\ Y_1^T = [1101000]\ and$$
$$y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_7 = 3\ \ i.e\ there\ exist\ a\ solution.$$

Then $p_i$, $p_k$ are in the same minimal P-smiflow or they are in two different minimal P-semiflow in (5). Now $LPP_2$ will determine which the case is. To the cycle **For** $LPP_2$ in (6), $\mathbf{Y_2^T}$=0 has the same equations than $Y_1^T$=0 in (5) but with the following conditions:

$$\boldsymbol{Y_2(y_2) = 0\ and\ Y_1(y_1) = 1}$$
$$\boldsymbol{The\ condition\ 0 \leq Y_2(p_r) \leq 1\ implies\ that}$$
$$\boldsymbol{0 \leq Y_2(y_3) \leq 1, 0 \leq Y_2(y_4) \leq 1, 0 \leq Y_2(y_5) \leq 1, 0 \leq Y_2(y_6) \leq 1,}$$
$$\boldsymbol{0 \leq Y_2(y_7) \leq 1.\ And\ the\ condition\ Y_2(i) \leq Y_1(i)\ \ implies\ that}$$
$$\boldsymbol{Y_2(y_1) \leq 1, Y_2(y_2) \leq 1, Y_2(y_3) \leq 0, Y_2(y_4) \leq 1,}$$
$$\boldsymbol{Y_2(y_5) \leq 0, Y_2(y_6) \leq 0, Y_2(y_7) \leq 0}$$
$$\boldsymbol{if\ y_1 = 1\ and\ y_2 = 0\ \rightarrow\ y_3 = 1, y_4 = 0, y_5 = 1, y_6 = y_7 = 0}$$
$$\boldsymbol{o\ y_6 = y_7 = 1\ but\ y_3 = 0\ \ because\ \ Y_2(y_3) \leq 0} \quad (6)$$

Thus, this value (6) does not fulfill that $-y_1+y_2+y_3=0$ so $Y^T_2$ = {} then $p_i$, $p_k$ are in the same minimal P-semiflow and the siphon $S_{ti}$= $\{p_2, p_1\}$ is formed. Now $LPP_1$ is executed to choose the following place $p_i$=$p_3$. **For** $LPP_1$:

$$For\ p_i \neq p_k, p_i = p_3\ and\ p_k = p_2, Y_1(y_2) = 1\ and\ Y_1(y_3) = 1.$$
$$The\ condition\ 0 \leq Y_1(p_r) \leq 1,\ implies\ that\ \ 0 \leq Y_1(y_1) \leq 1,$$
$$0 \leq Y_1(y_4) \leq 1, 0 \leq Y_1(y_5) \leq 1,\ \ 0 \leq Y_1(y_6) \leq 1, 0 \leq Y_1(y_7) \leq 1. \quad (7)$$
$$If\ y_3 = y_2 = 1\ \rightarrow y_1 = 2, y_4 = 1, and\ y_5 = 1,$$
$$if\ y_6 = 0 \rightarrow\ y_7 = 0\ \ but\ \ y_1 = 2\ \ not\ fulfills\ \ 0 \leq Y_1(y_1) \leq 1$$

Therefore, there is not a solution (7) then $Y^T_1$={} then $p_i$, $p_k$ are in different minimal P-semiflow. Now, $LPP_1$ is executed to choose the following place $p_i$=$p_4$. **For** $LPP_1$:

$$\boldsymbol{For\ p_i \neq p_k, p_i = p_4\ and\ p_k = p_2, Y_1(y_2) = 1\ and\ Y_1(y_4) = 1.}$$
$$\boldsymbol{The\ condition\ 0 \leq Y_1(p_r) \leq 1,\ \ implies\ that\ \ 0 \leq Y_1(y_1) \leq 1,}$$
$$\boldsymbol{0 \leq Y_1(y_4) \leq 1, 0 \leq Y_1(y_5) \leq 1,\ \ 0 \leq Y_1(y_6) \leq 1, 0 \leq Y_1(y_7) \leq 1.}$$
$$\boldsymbol{If\ y_4 = y_2 = 1\ \rightarrow y_3 = 0, y_1 = 1, y_5 = 0,\ if\ y_6 = 0 \rightarrow\ y_7 = 0}$$
$$\boldsymbol{then\ Y_1^T = [1101000]\ and}$$
$$\boldsymbol{y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_7 = 3\ \ i.e\ there\ exist\ a\ solution.}$$
(8)

Then $p_i$, $p_k$ are in the same minimal P-semiflow or they are in two different minimal P-semiflow (8). Now $LPP_2$ will determine which the case is. After executing the $LPP_2$ the value does not fulfill that $-y_2+y_4=0$ so $Y^T_2$ = {} then $p_i$, $p_k$ are in the same minimal P-semiflow and the siphon $S_{ti}$= $\{p_2p_1p_4\}$ is formed. Again, the $LPP_1$ is executed to choose the following place $p_i$=$p_5$ and there is not a solution then $Y^T_1$= {} then $p_i$, $p_k$ are in different minimal P-semiflow. Again, $LPP_1$ is

executed to choose the following place $p_i=p_6$ and $p_i$, $p_k$ are in the same minimal P-semiflow or they are in two different minimal P-semiflow. After, the $LPP_2$ is executed to determine which the case is. Then $p_i,p_k$ are in the same minimal P-semiflow or they are in two different minimal P-semiflow and again $LPP_2$ is executed. Thus, then $p_i,p_k$ are in different minimal P-semiflow and the $LPP_1$ is executed to choose the following place $p_i=p_7$. All the places were analyzed then go to end for. Then $S_{ti}= \{p_2p_1p_4\}$ Do $T^{ti}= \bullet S_{ti} \cup S_{ti} \bullet = \{t_1t_2t_4\}$.

Then, in cycle **For** $LPP_3$ the objective function is:

$$Max \sum_{i=1}^{[7]} Y_1(i) = Min\,(y_1+y_2+y_3+y_4+y_5+y_6+y_7) \;\; where$$

$$C=\begin{vmatrix} -1 & 0 & 0 & 1 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 \end{vmatrix}, X_1=\begin{vmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{vmatrix}, \begin{matrix} -x_1+x_4=0 \\ x_1-x_2=0 \\ x_1-x_3=0 \\ x_2-x_4=0 \\ x_3-x_4=0 \\ -x_4+x_5=0 \\ x_4-x_5=0 \end{matrix}, Z=\begin{vmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{vmatrix} \qquad (9)$$

$$\boldsymbol{Then\ x_1=x_2=x_4=x_3\ and\ x_4=x_5}.$$

$$\boldsymbol{The\ condition\ 0 \le X(h) \le 1,\quad implies\ that\ 0 \le x_1 \le 1,}$$

$$\boldsymbol{0 \le x_2 \le 1, 0 \le x_3 \le 1,\ 0 \le x_4 \le 1, 0 \le x_5 \le 1.}$$

$$\boldsymbol{and\ the\ last\ condition\ is\ Y_2(p_k)=0.}$$

$$\boldsymbol{The\ solution\ is\ X=[00000].}$$

Since X is not empty because $x_1+x_2+x_4=0$ and $X'Z=0$ then according to the $LPP_3$ in [10] this implies the IPN is not diagnosable. However, it is noticed that with the condition given above, for this IPN, X always will have a solution because $x_1=x_4=x_2=x_3$ and $x_4=x_5$, and $x_1+x_2+x_4=0$. Since X cannot be negative, the only possible result is that $x_1=x_4=x_2=x_3= x_5=0$, *i.e.*, $X= [00000]$. The proposal in [10] states that X must be empty when the IPN is diagnosable, but X cannot be empty with the conditions given $x_1+x_2+x_4=0$ and $x_1=x_4=x_2=x_3=0$, so X has a solution and the solution is equal to the zero vector. Thus, if X has a solution and this solution is zero vector $X= [00000]$ then $t_i$ will be diagnosable. This is considered in the implementation section. Furthermore, the condition $Y_2(p_k)=0$ is no necessity in the $LPP_3$ because is maintained when is used the $LPP_2$.

### 4.3. Implementation

The algorithm is divided into two parts; the first part (DiagFaultTi1) contains the algorithms $LPP_1$ and $LPP_2$ that determine if there exist a siphon that cannot be marked when a fault occurs:

```
> DiagFaultTi1:=proc(C,pk,k,au)
#The numbers of columns and rows from C is calculated
>colc:= linalg:-coldim(C): rowc:+linalg:-rowdim(C)
#The P-semiflow vector Y1 and Y2 is definted
>Y1y:=Vector[linalg:-row](rowc,0);
>for i to rowc do Aux1:=cat(y,i); Y1y[i]:=Aux11 end do;
>Y1y; Y2y:=Y1y;
# The objective function of LPP1 and LPP2 is definited
>Calobi1 := linalg:-vector(rowc, 0); Y1ecv:={}; Y2ecv:={} end do
>print ("Siphon that not will be market when a fault occurs"); print (Sti);
# The P-semiflow and the Sti is calculated
>P:=Vector[linalg:-row](rowc,0)
> for i to rowc do aux2:=cat(p,i); P[i]:=aux2 end do; P; Sti:={pk}
# The conditions s.t. from the objective funtion are calculated
>Cond1:='union'('union'('union'(Y1e0,Y1ecv),{Y1a[k]}),{Y1a[i]});
>Y1:=minimize(Obj1,Cond1,NONNEGATIVE);
>Cond2:='union'('union'('union'(Y2e0,Y2ecv),{Y2a[k]}),{Y2a[i]});
# the objetive function of LPP2 is calculated
>Y2:=minimize(Obj2,Cond2,NONNEGATIVE);
if Y2={} the Sti:='union'(Sti,{pj}) else Y2t:=table(Y2);
Y2ms:=convert(Y2t, multiset); Y1ecv:={}; Y2ecv:={} end do
>print ("Siphon that not will be market when a fault occurs"); print (Sti);
```

The second algorithm (DiagnFaultTi2) contains the algorithm $LPP_3$ that determine if the net is diagnosable with the output of the first algorithm:

```
> DiagFaultTi2:=proc(C,Ttiv,num,au)
>colc:= coldim(C);X1x:=Vector(colc,0);X1vc:=Transpose(X1x);
>for i to colc do Aux2:=cat(x,i); X1x[i]:=Aux2 end do;
>X1e:=multiply(C,X1x); X1e0:=equate(X1e,0); X1ecv:={};
>CalobjX1:=vector(colc,0); for i to colc do CalobjX1[i]:=X1x[i] end do;
>ObjX1:=multiply(M,CalobjX1); X1T:=Transpose(X1x);
>Z:=Vector(colc,0); T: =Vector [row](colc,0);

>for j to colc do aux3:=cat(t,j); T[j]:=aux3 end do; Tti:={};
>for j to num do for i to colc do if Ttiv[j]=T[i] then Z[i]:=1 end if
> end do end do;
>CondX1:= ('union'('union'(X1e0,X1ecv),{X1TZe0});
> X1:=maximize(ObjX1,CondX1,NONNEGATIVE);
>NZ:=ArrayNumElems (X1aa,NonZero);
>if NZ=0 then print("The fault is Diagnosable") else
> print ("The fault is No diagnosable") end if end proc
```

## 5. CONCLUSIONS

The definition, analysis and the implementation of the algorithm proposed by [10] to detect faults was presented. It can be observed that the condition to apply the algorithm to diagnose the IPN is that X cannot be empty, and the solution must be zero vectors to the fault $t_i$ can be diagnosed.

In the future, is considered that these three algorithms can be executed together, integrating the LPPS presented. Also, will be explored the possibility of its implementation in a Programmable Logic Circuit that can be embedded in a DES.

## 6. REFERENCES

[1] S. Lafortune, D. Teneketzis and M. Sampath, "Failure Diagnosis of Dynamic Systems: An approach based on Discrete Event Systems", Proceedings of the American Control Conference, 2001 pp. 2058-2068.

[2] M. Silva, Las redes de Petri: en la automática y la informática. Editorial AC, Madrid, España, 1985, pp. 558-562.

[3] T. Ushio, I. Onishi and K. Okuda, "Fault Detection Based on Petri Net Models with Faulty Behaviors". IEEE Transactions on Automatic Control. Vol. 50, no. 4, 1998 pp. 476-492.
[4] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen and D. Teneketzis, “Failure Diagnosis Using Discrete-Event Models”, IEEE Transactions on Control Systems Technology. Vol. 4, No. 2, 1996, pp.105-124.
[5] D. De Tommasi, F. Basile, and P. Chiacchio. "Sufficient conditions for diagnosability of Petri Nets". Proceedings of the 9th International Workshop on Discrete Event Systems Göteborg, Sweden, May 28-30, 2008, pp. 370-375.
[6] D. De Tommasi, F. Basile, and P. Chiacchio. "An efficient approach for on-line diagnosis of discrete event systems". IEEE Transactions on Automatic Control, 54(4):74-759, April 2009.
[7] M. Dotoli, M.P. Fanti, A.M. Mangini and W. Ukovich, “On-line Fault Detection in DES by Petri nets and Integer Linear Programming”. Automatica. Vol. 45. no.11, 2009, pp. 2665-2672.
[8] M. Cabasino, A. Giua, S. Lafortune, and C. Seatzu. “Diagnosability analysis of unbounded petri nets”. Proceeding of the 48th IEEE Conference in Decision and Control 2009, pp. 1267-1272. doi:10.1109/CDC.2009.5400608.
[9] A. Ramírez-Treviño, E. Ruiz-Beltrán, J. Arámburo and E. López-Mellado. Structural Diagnosability of DES and Design of Reduced Petri Net Diagnosers. IEEE Transaction on Systems, MAN and Cybernetics. Vol. 42, No. 2, 2012, pp. 416-429.
[10] E. Ruiz-Beltrán, A. Ramirez-Treviño and J.L. Orozco-Mora. "Fault Diagnosis in Petri Nets". Formal Method in Manufacturing. CRC Press Taylor-Fracis Group, Boca Raton, FL. 2014. pp 728.
[11] C.G. Cassandras and S. Lafortune, Introduction to Discrete Event Systems, Editorial Springer. Second edition. New York, USA, 2008.
[12] K. Hernández-Rueda and M. E. Meda-Campaña, “Fault Diagnosis in Discrete Events Systems Not Diagnosable”. MemoriaElectro 2015. Vol. 37, pp.141-146.
[13] A. Ramírez-Treviño, E. Ruiz-Beltrán, I. Rivera-Rangel and E. López-Mellado, “Online Fault Diagnosis of Discrete Event System. A Petri Net Based Approach”. Transaction on Automation Science and Engineering Vol. 4, no. 1, 2007, pp. 31-39.