Diagnosis of Intermittent Faults and its dynamics

A. Correcher, E. García, F. Morant, E. Quiles and L. Rodríguez
Universidad Politécnica de Valencia
Spain

1. Introduction

Intermittent faults (IFs) are difficult to diagnose and may cause a great disruption in industrial processes. Most IFs are related to gradual degradation of components or systems. For instance, evolution of connection failures is shown in Fig. 1 (Correcher et al., 2004), (Sorensen et al, 1998). Connection failures are rarely repaired so its behaviour worsens over time. Intermittent faults behave as small noise fluctuations in stage 1 of their development. As the amplitude and duration of the fluctuations increase (stage 2), IFs start to occur. The effects of IFs are severe in stage 3.

Therefore, in many instances, the occurrence of IFs in a device is a prelude of permanent failures (PFs). In these cases, if IFs can be detected then appropriate actions could be taken in order to minimise the economic impact.

In (Correcher et al., 2004) an IFs diagnosis tool was presented by the authors. This tool was able to diagnose the failure and recovery events in a system with IFs. This paper presents an extension of the work in (Correcher et al., 2004) which includes not only event detection but also fault dynamics detection, defined as the evolution of IFs occurrence over time. Other approaches to IFs diagnosis (Contant et al., 2004), (Jinag et al. 2003), do not consider IF dynamics.

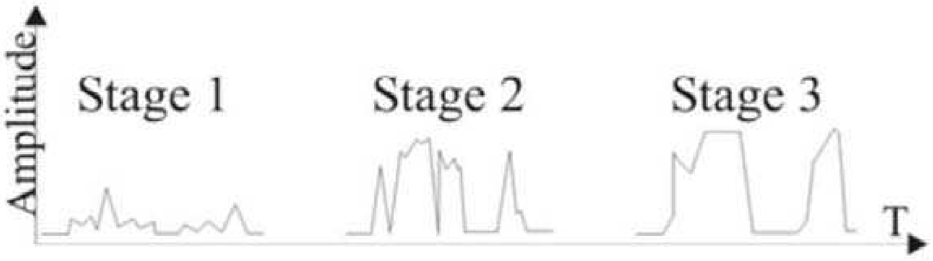


Fig. 1. Connection IF evolution through device life.

However, the existence of IF dynamics is experimentally shown in (Sorensen et al., 1998) and in the destructive tests presented in this paper. Therefore, we can introduce the diagnosis problem to be addressed.

Definition 1: IF diagnosis problem.
Starting from a temporal input *(U)* and output *(Y)* sequence, obtained by means of sensors in the process, compute the presence of any failure *f* (with *f* included in a failure set), its recoveries and its dynamics.

The proposed solution has a clear industrial interest, as not only diagnoses IFs, but also provides valuable information on the fault evolution for maintenance purposes. First, section 2 presents a study about the evolution of the IF during its online diagnosis. This approach is able to extract some characteristic parameters of the IF. These parameters will be useful for estimating the behaviour of the fault in the future. The approach is also applied to experimental data.

Section 3 analyses the effectiveness of the approach in the solution of the problem stated in definition 1 when diagnosing Discrete Event Systems (DES) (Cassandras & Lafortune, 1999). Section 4 presents an application of IF dynamics diagnosis with Coloured Petri Nets. Finally, we present some conclusions in section 5.

2. Temporal modeling

IF dynamics characterization generates useful information for preventive maintenance scheduling. Two complementary parameters are defined in this section: temporal failure density (DF) and pseudoperiod *(Ps)*. The goal of these parameters is to characterize the IF dynamics. *DF* and *Ps* can be on-line computed. *DF* and *Ps* can also predict future behaviour of the faulty device.

DF and *Ps* are computed from IF time occurrence and IF duration (defined as the time difference between fault and recovery). Therefore, two arrays are computed for each fault: the fault time vector ($FT_{Fj}=[FT_{(1)Fj},FT_{(2)Fj},\ldots,FT_{(n)Fj}]$) and the duration time vector ($T_{Fj}=[T_{(1)Fj}, T_{(2)Fj},\ldots,T_{(n)Fj}$]), where F_j stands for a fault included in the fault set and "*n*" is the index number of detected faults. Arrays and parameters are computed recursively on-line considering a moving time window of a given duration.

2.1 Temporal failure density

Temporal failure density *(DF* or density in the rest of the paper) is defined as the average time a particular fault (F*j*) is active within a sliding time window of duration *W*. Therefore, if we define the current time as "*ti*", the density is computed from "*ti-W*" to "*ti*". Therefore, *DF* is computed for time "*ti*" as:

$$DF_{ti} = \frac{\sum_{i=k}^{CNT}\left(T_{(i)Fj} - T_A\right)}{W}; CNT > 0 \tag{1}$$

where *CNT* is the number of faults inside the window, "*k*" stands for the index of the first fault detected inside the window {*k*: *FT(k)Fj ≥ (ti-W)* }(k: $FT_{(k)Fj}$ > *(ti-W)* and $FT_{(k-1)Fj}$<*(ti-W)*} if exists, otherwise *{k=CNT+1}* and "T_A" takes into account the duration of a fault occurred before "*ti-W*" which continues active inside the window. Therefore:

$$T_A = FT_{(k-1)Fj} + T_{(k-1)Fj} - (ti - W) \tag{2}$$

Equation (2) is valid only if "T_A" is positive, otherwise "$T_A=0$", as this fact would indicate that the first considered fault is completely outside of the window.

In a real system, *DF* tends to increase with time; thus confirming the hypothesis that IFs progressively damage the faulty device. Figures 2 and 3 show the computed *DF* from experimental data and its filtered signal (low-pass fourth order Butterworth filter). The experimental data has been obtained from ten million operation tests on relays switching a resistive load. As the time between operations was 100 milliseconds, the overall duration of each experiment was 277.8 hours.

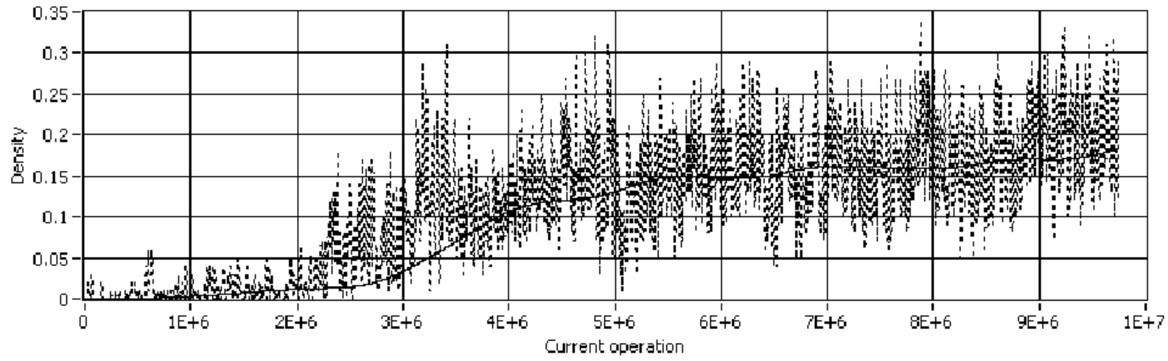


Fig. 2. Fault density. Window size is 10000 operations.

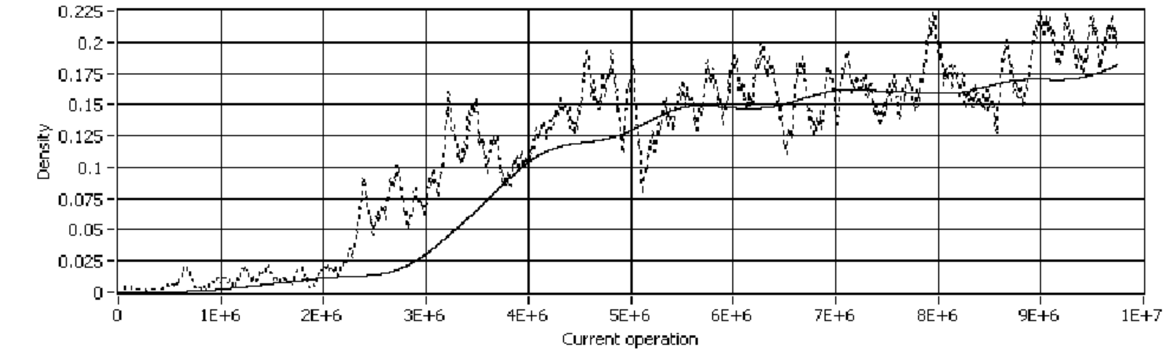


Fig. 3. Fault density. Window size is 100000 operations

The rising characteristic of the density can be used to estimate the optimal time to repair or substitute the faulty device. Effectively, a certain maximum density threshold can be defined as the limit for unacceptable behaviour. Then, an adequate extrapolation model can be used to predict the number of operations the device is capable of carrying out before reaching the unacceptable behaviour limit. Obviously, the unacceptable density threshold should be defined specifically for each process device, depending on its specific functionality and reliability requirements.

The simplest prediction model consists of a density linear increase. In this case, filtered density data can be treated with classical techniques (Lemmis, 1995) such as least squares (LS) or recursive least squares *(RLS)*. Therefore, we obtain a model:

$$D = m_{ti} \bullet t + n_{ti} \tag{3}$$

where D stands for the density, t stands for the time and the subindex ti stands for the time value when density is estimated. If we consider a threshold "D_0", the time (t_{D0}) when the density "D_0" will be reached is:

$$t_{D0} = \frac{\left(D_0 - n_{ti}\right)}{m_{ti}} \tag{4}$$

Therefore, "t_{D0}" is the time when the faulty device should be replaced. This time is named as Linear Substitution Time at time "ti" (LST_{ti}). In addition, it is possible to define another parameter much more suitable for preventive maintenance: Operations to Replacement at time "ti" (OTR_{ti}). This parameter represents an estimation of the useful operations left on a device, and can be computed as:

$$OTR_{ti} = LST_{ti} - ti \tag{5}$$

Obviously, only positive values of OTR_{ti} are meaningful, otherwise the corresponding OTR_{ti} is considered equal to zero.

The proposed linear prediction model has been found to be suitable for the use with the experimental used through the chapter. However, this kind of model might not be adequate for other devices. For instance, if the fault density follows first order dynamics then *LSTti* will predict an optimistic substitution time. This problem can be solved by using *RLS* with forgotten factor to fit LST_{ti}. In any case, LST_{ti} and OTR_{ti} reflect the underlying fault dynamics, and could be used to model systems that do not follow linear increase laws.

Therefore, the fault diagnosis system will predict the time when a device must be replaced in two stages. First, fault density will be computed and, from this value, LST_{ti} and OTR_{ti} will be predicted.

As mentioned before, the sliding window size should be appropriately chosen, as short windows will imply high variability and noise in the calculated failure density and long windows, which exhibit a greater filtering effect, involve high computational costs and might mask part of the fault underlying dynamics.

Figures 2 and 3 show the effect of window size in the variability of the density. From calculations with a range of window sizes, it has been found that windows greater than 5000 operations include the same low frequency component, as shown in figs. 2 and 3. Therefore, LST_{ti} and OTR_{ti} computed from window sizes greater than 5000 operations are identical. Figure 4 shows the evolution of OTR_{ti} obtained from the experiments in figs 2 and 3. Figure 4 shows that the device should be replaced when reaching 6 million operations instead of the substitution time recommended by the manufacturer (10 million operations).

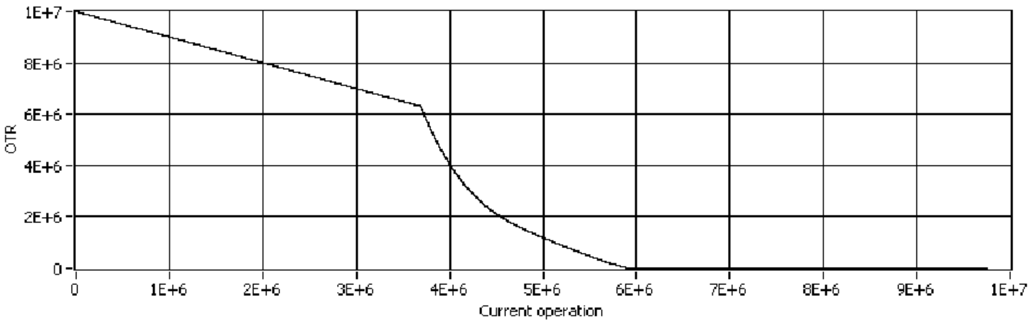


Fig.4. OTR_{ti} for an acceptable density threshold below 15%.

2.2 Pseudoperiod.

Fault density can be used to predict the device substitution time, however it does not completely explain IF dynamics. For instance, fig. 5 shows two cases with exactly the same failure density. However, the effects on the device are clearly different.

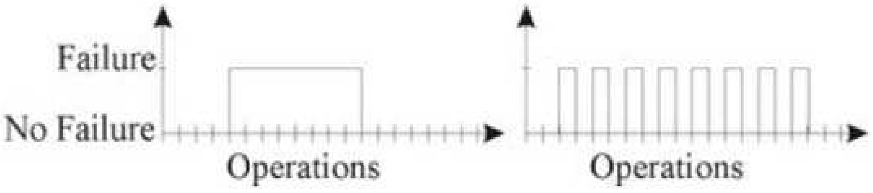


Fig. 5. IF with the same density but different dynamics.

The difference between the two behaviours can be modelled by the time difference between the occurrences of two consecutive faults.

The time difference can be measured with a new parameter, the Pseudoperiod *(Ps)*. *Ps* is defined as the average time difference between faults inside a sliding window. Moreover, *Ps* is normalized by the number of faults, and can be computed at time "*ti*" as:

$$Ps_{ti} = \frac{\sum_{i=j}^{i=k} FT_{i+1} - FT_i}{k - j + 1} \qquad (6)$$

where "ti" stands for current time, "*FT*" is the detection time for fault "*i*", and "*j*", "*k*" are the first and last fault indexes in the window, respectively.

Pseudoperiod is clearly a magnitude related to mean time between failures *(MTBF)*, commonly used to model reliability of reparable systems. Moreover, pseudoperiod is a dynamic magnitude. Therefore, we can compute a *Ps* curve for any IF. This curve can be used to predict the substitution time of the device.

The evolution of Pseudoperiod (figs. 6 and 7) shows an increase until a maximum value is reached, to decrease towards a value close to zero. This dynamic behaviour is consistent with the nature of IFs (fig. 1). The computed Pseudoperiod remains in the range 600 to 800 from 4 million operations onwards. Therefore, it is possible to conclude that, in average, the number of failures from the 4 millionth operation remains reasonably constant. Moreover, the average duration of each failure slowly increases with the number of operations, as the density (fig. 3) increases.

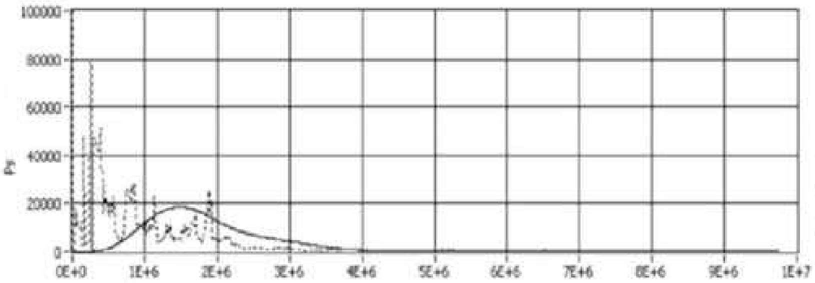


Fig. 6. Pseudoperiod. Window size is 100000 operations.

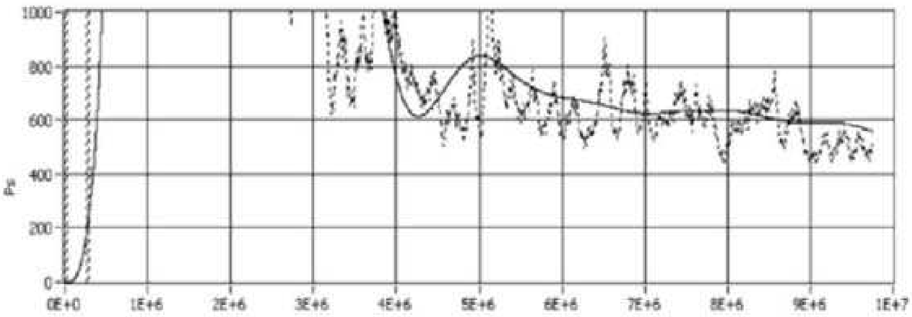


Fig. 7. Zoom in Pseudoperiod. Window size is 100000 operations.

Pseudoperiod can be used to compute some limit in the desirable behaviour of the system. The shape of the Pseudoperiod signal suggests the estimation of this limit with *RLS* with forgetting factor. Another solution could be a mixed model with a polynomial and linear estimation for each side of the signal. This last approach seems to be more promising but its computing is no trivial. Continuity and derivability must be guaranteed. Moreover, the order of the filter used in the Pseudoperiod signal will affect in the delay of the model. These problems will be addressed in future works.

3. Intermittent fault dynamics diagnosability

It is necessary to ascertain if the proposed dynamic parameters can be used to perform the complete fault diagnosis as per definition 1. To complete definition 1, a definition of fault dynamics diagnosis is introduced, based on the definition of discrete-time systems observability (Smolensky et. al, 1996) "A discrete-time system is observable if a finite *k* exists such that knowledge of the outputs to *k-1* is sufficient to determine the initial state of the system. "

Definition 2. IF dynamics diagnosis problem.

Given a temporal fault sequence, $TF = \{tf_i\} \ (i = 0...n)$, and a temporal recovery sequence, $TRF = \{trf_j\}(j = 0...m)$, compute the next value in TF if $m \geq n$ or the next value in TRF if $m < n$.

Definition 2 states that IF dynamics is diagnosable if we can compute the next time when a fault or recovery will occur. IFs are asynchronous and non-deterministic. So, IF dynamics cannot be diagnosed with deterministic precision. Therefore, we propose a relaxed definition.

Definition 3. Bounded IF dynamics diagnosis problem.

Given a temporal fault sequence, $TF = \{tf_i\} \; (i = 0...r)$, and a temporal recovery sequence, $TRF = \{trf_j\} \; (j = 0...s)$, compute the next value in $TF \; if \; s \geq r$ or the next value in $TRF \; if \; s < r$, with a bounded uncertainty.

This uncertainty can be used to compare different methods of diagnosis.

3.1 Bounded IF dynamics diagnosis with DF_{Tj}.

IF dynamics diagnosis start from a temporal fault event sequence and a temporal recovery event sequence until the actual time $\tau \; (TF = \{tf_i\}(i = 0...s), TRF = \{trf_j\}(j = 0...r))$. DF can be computed as in equations (1) and (2). Let us assume that $T_A=0$, therefore, the next event will be a fault. In this case, the problem in definition 3 will consist of computing the next fault time. As historical data is known until time τ, it is possible to compute a sequence of fault densities *$\{D_l\}(l=0... \tau)$.* Density RLS prediction computes the estimated density for a given instant of time υ: $D_\upsilon = m \bullet \upsilon + n$, where m and n are the results of the RLS estimation. Density increases when there is a new diagnosed fault in the system. Moreover, density increases for a maximum of "t_m/w", where "t_m" is the sampling period and "w" is the window size. Therefore, the next failure will occur when the linear model will estimate this density value.

$$D_\tau + \frac{t_m}{w} = m \bullet TF_{s+1} + n \tag{7}$$

So

$$TF_{s+1} = \frac{w \bullet (D_\tau - n) + t_m}{w \bullet m} \tag{8}$$

The estimation error is therefore, the error in the RLS estimation (Smolensky, et. al, 1996). If we want to compute the time for a recovery, ($T_A \neq 0$) the density will decrease in, at least, "t_m/w", so:

$$RTF_{r+1} = \frac{w \bullet (D_\tau - n) - t_m}{w \bullet m} \tag{9}$$

We can conclude that the density allows for the complete diagnosis of the IF dynamics with a bounded error.

In order to compute the IF density, the fault diagnosis system should include the identification of fault starting time and duration. The fault diagnosis system should also be able to compute the corresponding fault densities.

4. Latent Nestling.

The previous section showed that the diagnosis of an IF involves not only the diagnosis of fault and recovery events, but also the diagnosis of its dynamics. This section shows how to diagnose IF dynamics with the methodology based on Coloured Petri Nets (CPN) presented in (Garcia et al., 2008) (Rodriguez et al., 2008). This methodology allows IF dynamics diagnosis because it includes timing information. We also include a complete diagnosis example of an industrial process.

A Coloured Petri Net for Fault Diagnosis (DCPNs) is:

$$D=(P,T,Post,M_0,C,f,PLNf,T_f,PVf) \tag{10}$$

where

- P is a finite set of places.
- T is a finite set of transitions.
- Pre and $Post$ are input and output arc functions.
- M_0 is the initial marking.
- C is the colour set assigned to different identifiers. $C = N \cup f$ is the subset of coloured tokens representing the normal system behaviour.
- $f = \{f_1, f_2, ..., f_i\}$ is the subset of coloured tokens representing fault set.
- $PLNf \subseteq P$ is the subset of fault latent nestling places.
- $PVf \subset P$ is the subset of fault verification places.
- $T_f \subset T$ is the transition subset including coloured functions. T_f links $PLNfi$ with $PVf.$

Fault verification places are P-timed. Therefore, they include pairs <R, Tim_R>*;* where R is a coloured mark and Tim_R is a timer. Tim_R will be used to compute IF Density and Pseudoperiod.

Thorough this paper the notation $M(Pl(\langle V\rangle))$ {Pl ∈ P; V ∈ C} will refer to the marking of place "Pl". This notation represents that there is a "V" coloured token in place "Pl" (for Pre functions). This notation also represents that a "V" coloured token is placed in "Pl" (for $Post$ functions).

4.1 System modelling

The first step of the method consists of the dynamic system modelling. The system model is designed with generalized Petri Nets (PNs) (David & Alla, 2005), for simple systems, or with CPNs (for complex systems) (Hensen, 1997).

Let us show the methodology applied to a rectifying industrial machine. The machine rectifies blocks of a synthetic compound that imitates the natural stone. Figure 8 shows the process scheme. The rectifying process consists of the mechanical elimination of some material in order to achieve the desired width. The system can be divided into four subsystems. Since each milling works with an independent motor, each subsystem will consist of a pair motor-milling with a blade cooling and lubrication system. The blade cooling system consists of a pair pump-valve that pours cutting oil over the millings. The motors can suffer Ifs resulting from fretting corrosion in their electrical contacts. The millings fail when there is any defect in the tool. This failure is a PF that can be due to maintenance failure. Milling failure will cause a great torque and, therefore, a power consumption greater that usual.

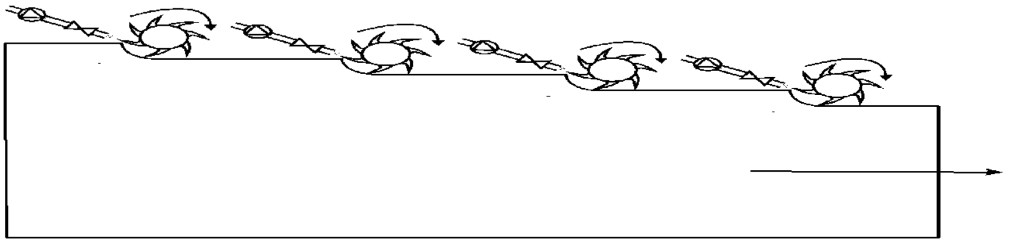


Fig. 8. Artificial Stone rectifying process.

Moreover, a fault in a previous subsystem can cause the same symptoms because the milling will cut more material. The cooling and lubrication system can also suffer IFs. Typical IFs in a pump-valve system are electrical contact failures and valve blocking.
Figure 9 and table 1 show the PN system model. Table 2 shows the relationship between places and system states.

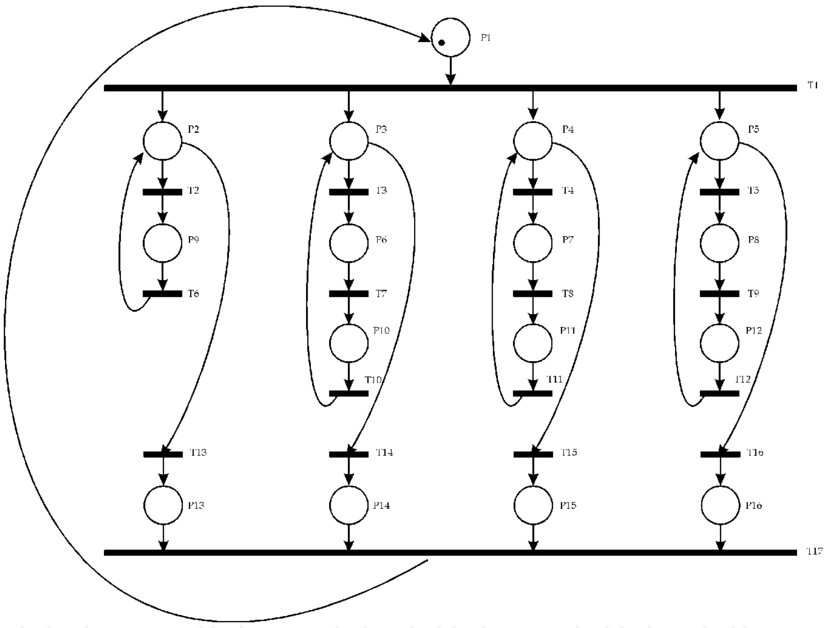
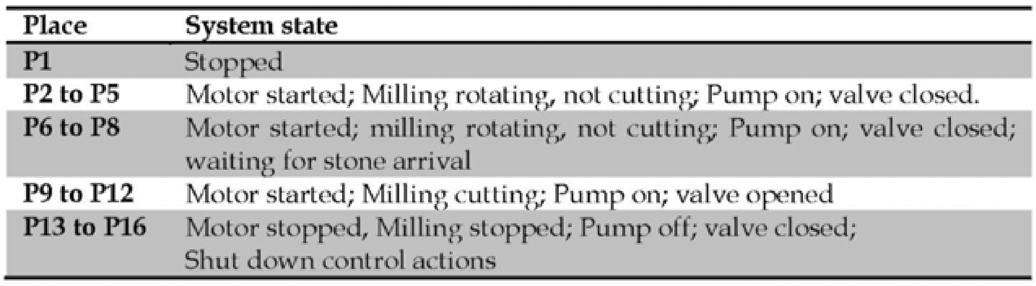


Fig. 9. System PN model.

Tr	Event	Tr	Event	Tr	Event
T1	Start	**T7**	New stone	**T13**	Stop
T2	New stone	**T8**	New stone	**T14**	Stop
T3	*t2/p3/#1s*	**T9**	New stone	**T15**	Stop
T4	*t3/p4/#2s*	**T10**	*tp2/p9/#10s*	**T16**	Stop
T5	*t4/p4/#3s*	**T11**	*tp3/p9/#10s*	**T17**	Stop
T6	*tp1/p9/#10s*	**T12**	*tp4/p9/#10s*		

Table 1. Transitions in PN system model.



Place	System state
P1	Stopped
P2 to P5	Motor started; Milling rotating, not cutting; Pump on; valve closed.
P6 to P8	Motor started; milling rotating, not cutting; Pump on; valve closed; waiting for stone arrival
P9 to P12	Motor started; Milling cutting; Pump on; valve opened
P13 to P16	Motor stopped, Milling stopped; Pump off; valve closed; Shut down control actions

Table 2. Places in PN system model.

The next step consists of the folding to a CPN (Garcia et al., 2008). Figure 10 shows the result. The CPN system model stars in *Pc1*. *Tr1* starts all subsystems and generates normal working tokens in *Pc2*. Arc functions *g* and *g1* denote the relationship between the general normal working token with particular normal working tokens:

$$\left[g\left(\langle \bullet n \rangle\right) = C \right.$$

$$\left[g1\left(\langle C \rangle\right) = \bullet_n \right.$$

$$C = \langle S1_n \rangle + \langle S2_n \rangle + \langle S3_n \rangle + \langle S4_n \rangle$$

where $Sj_n \left(j \in \{1..4\} \right)$ stands for normal subsystem coloured token.

Transition *Tr2* puts a stone in the machine. This action starts the first subsystem (place *Pc4*) and moves the other three subsystems to "waiting for stone arrival" state (*Pc3*). Function *gs1* and *gsk* model the different paths followed by subsystems:

$$\left[gs1\left(\langle S1_n \rangle\right) = \langle S1_n \rangle \right.$$

$$\left[gsk\left(\langle S2_n \rangle + \langle S3_n \rangle + \langle S4_n \rangle\right) = \langle S2_n \rangle + \langle S3_n \rangle + \langle S4_n \rangle \right.$$

Transition $Tr3 = \{T7, T8, T9\}$ starts sequentially the other subsystems by marking *Pc4*. Transition $Tr4 = \{T6, T10, T11, T12\}$ stops sequentially the cutting process of each subsystem. Transition *Tr5* starts the shutdown routine for every subsystem.

4.2 Fault set definition

The next step is the fault analysis of system devices. The goal is to define the faults to be diagnosed in each device. Each fault has a coloured fault token. Therefore, the fault set consists of the union of the coloured fault tokens $f = \{f_1, f_2, \quad , f_i \}$. Fault isolation will be guaranteed because any fault is associated to a device. Each subsystem includes four devices: motor, milling, pump, and valve. Table 3 shows the faults included in the example:

Fault	Definition	Effect	Nature
f_1^j $j \in \{1..4\}$	Subsystem *j* Motor fault. Fault in the electrical contacts	Motor stopped	Intermittent
f_2^j $j \in \{1..4\}$	Subsystem *j* Milling fault.	Milling cuts less material	Permanent
f_3^j $j \in \{1..4\}$	Subsystem *j* Pump fault. Fault in the electrical contacts	Pump stopped	Intermittent
f_4^j $j \in \{1..4\}$	Subsystem *j* valve fault.	Valve blocked in opened position	Intermittent
f_5^j $j \in \{1..4\}$	Subsystem *j* valve fault.	Valve blocked in closed position	Intermittent

Table 3. Fault set definition

4.3 Places of latent nestling faults, *PLNf*

The next step consists of marking all fault coloured tokens in specific CPN places. These places are called "Places of latent nestling faults" *(PLNf)*. Expert's empirical knowledge sets the rules for this operation. Figure 10 shows the marking for this example.

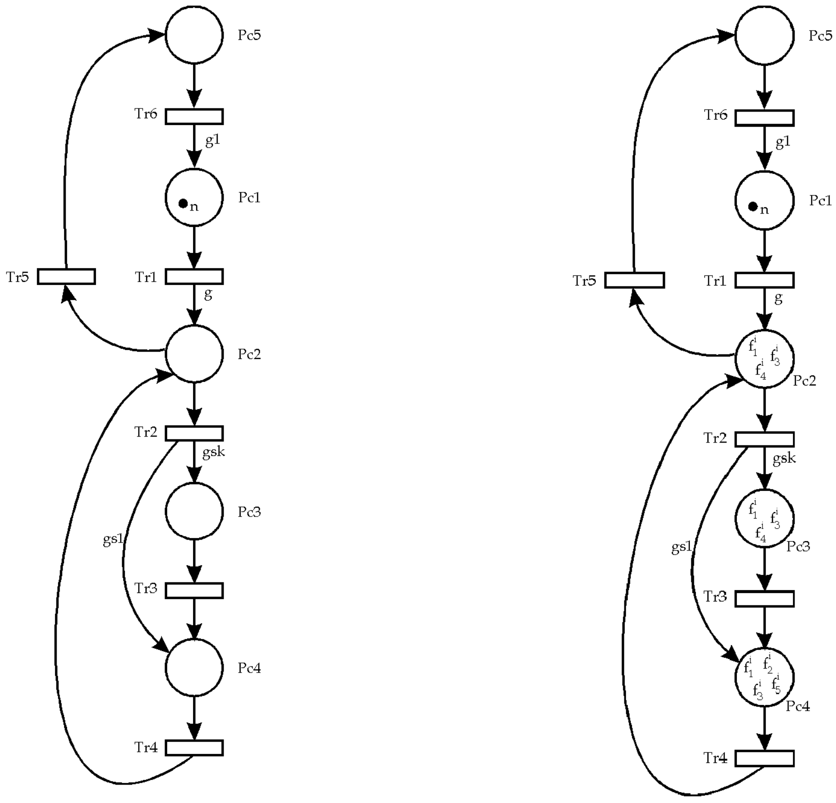


Fig. 10. System CPN model and fault allocation $(j=\{1,2,3,4\})$.

The computing of the thresholds to generate *k(N)* and $I_i(S)$ events is not trivial, because the sensor will observe an overcurrent when the tool touches the block (normal working). Nevertheless, we can easily generate these events if having a normal working current pattern. Let us suppose that we have a current pattern for each motor: $Pat_i(t)$. The continuous measure of the sensor is $I_i(t)$. Therefore, for each time t_k:

$$\left|Pat_i(t_k) - I_i(t_k)\right| < d_{\max} \Rightarrow \textit{normal working}$$
$$\left|Pat_i(t_k) - I_i(t_k)\right| > d_{\max} \Rightarrow \textit{faulty working}$$

where d_{max} is the maximum difference allowed between both signals. Therefore, we can define the three current events as:

$$\left|Pat_i(t_k) - I_i(t_k)\right| < d_{\max} \; and \begin{cases} \left|I_i(t_k)\right| < \gamma \Rightarrow I_i(0) \\ \left|I_i(t_k)\right| > \gamma \Rightarrow I_i(N) \end{cases}$$
$$\left|Pat_i(t_k) - I_i(t_k)\right| > d_{\max} \; and \begin{cases} \left|I_i(t_k)\right| < \gamma \Rightarrow I_i(0) \\ \left|I_i(t_k)\right| > \gamma \Rightarrow I_i(S) \end{cases}$$

where y is close to zero and it allows noise filtering.

Flow sensors, Fl_i ($i\in$\{1..4)\}) generate two levels: $Fl_i(0)$, no flow; $Fl_i(1)$, flow. Pressure sensors, Pr_i ($i\in$\{1..4)\}) generate two levels: $Pr_i(0)$, no pressure; $Pr_i(1)$, pressure.

The set of sensor values is *SM*. The set of possible sensor values combinations for marking M_k is denoted as "$SROVj(M_k)$" ($j\in\{1...n)\}$), where "n" is the number of possible combinations. "$SROVj(M_k)$" can be split into two subsets: $SROV(M_k)=SROV_{ev}(M_k) \cup SROV_{nev}(M_k)$. "$SROV_{ev}(M_k)$" stands for the subset of expected values and "$SROV_{nev}(M_k)$" stands for the subset of non expected. We use "$SROV_{nev}(M_k)$" to generate the trajectories of fault verification.
We use this information to build the system diagnoser. The diagnoser consists of the extension of the system model by including the trajectories of fault verification. The complete algorithm is shown in (Garcia et al., 2008). Figure 12 shows the CPN diagnoser. Figure 12 duplicates places *Pc2, Pc3,* and *Pc4.* Nevertheless, the diagnoser includes only one place *Pc2,* one *Pc3,* and one *Pc4.* We use the representation in figure 12 to increase the readability of the figure.
Table 4 defines the trajectories of fault verification for the diagnoser. "*" stands for any value of other sensors. Function *g2* evaluates place markings and returns fault marks when necessary.

$g2(Pl,V)\ \{Pl \in P; V \in f\}$:
$if\ M(Pl(\langle V \rangle))\ then\ null\ else\ M(Pl(\langle V \rangle))$
The diagnoser must solve the problem of chained faults. The main effect of any subsystem fault is less material cut. Therefore, the following subsystem will observe greater currents. Nevertheless, this situation does not involve two faults. *Tf3* (table 4) solves the problem by including previous subsystem faults in the diagnosis.

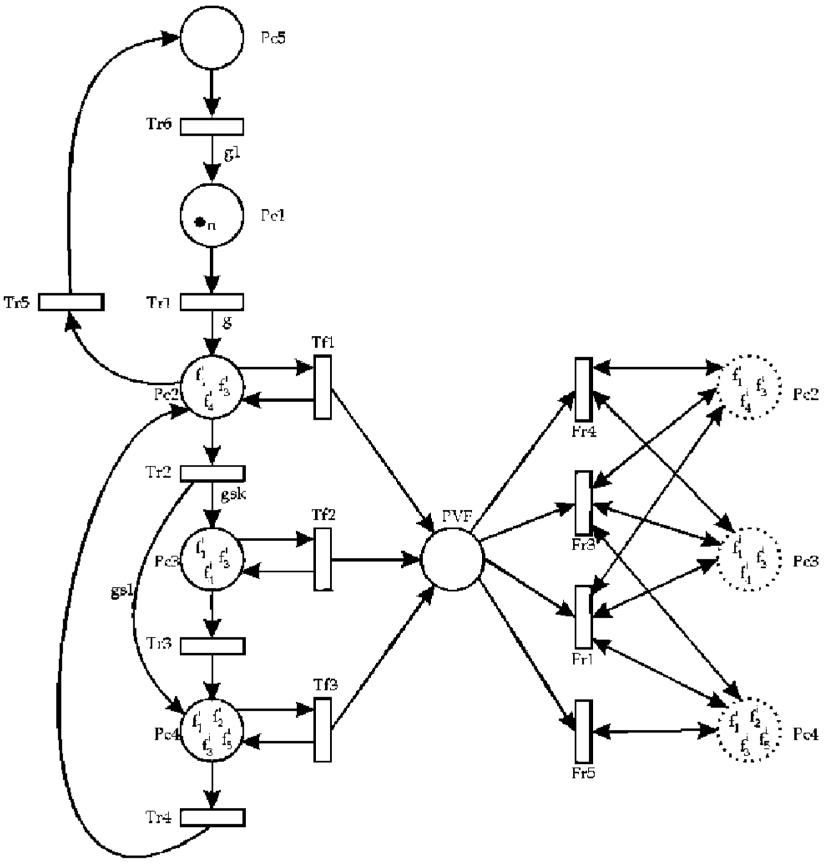
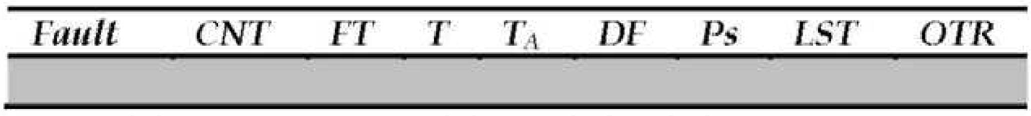


Fig. 12. Diagnoser

T	*Pre*	*Sensors*	*Post*	*j*
Tf1	$M(Pc2(\langle Sj_n\rangle,\langle f_1^j\rangle))$	$I_j(0)$,*	$M(Pc2(\langle Sj_n\rangle))\wedge M(PVF(\langle f_1^j\rangle))$	(1,2,3,4)
	$M(Pc2(\langle Sj_n\rangle,\langle f_4^j\rangle))$	$Fl_j(1)$,*	$M(Pc2(\langle Sj_n\rangle))\wedge M(PVF(\langle f_4^j\rangle))$	(1,2,3,4)
	$M(Pc2(\langle Sj_n\rangle,\langle f_3^j\rangle))$	$Pr_j\ (0)$,*	$M(Pc2(\langle Sj_n\rangle))\wedge M(PVF(\langle f_3^j\rangle))$	(1,2,3,4)
Tf2	$M(Pc3(\langle Sj_n\rangle,\langle f_1^j\rangle))$	$I_j(0)$,*	$M(Pc3(\langle Sj_n\rangle))\wedge M(PVF(\langle f_1^j\rangle))$	(2,3,4)
	$M(Pc3(\langle Sj_n\rangle,\langle f_4^j\rangle))$	$Fl_j(1)$,*	$M(Pc3(\langle Sj_n\rangle))\wedge M(PVF(\langle f_4^j\rangle))$	(2,3,4)
	$M(Pc3(\langle Sj_n\rangle,\langle f_3^j\rangle))$	$Pr_j\ (0)$,*	$M(Pc3(\langle Sj_n\rangle))\wedge M(PVF(\langle f_3^j\rangle))$	(2,3,4)
Tf3	$M(Pc4(\langle Sj_n\rangle,\langle f_1^j\rangle))$	$I_j(0)$,*	$M(Pc4(\langle Sj_n\rangle))\wedge M(PVF(\langle f_1^j\rangle))$	(1,2,3,4)
	$M(Pc4(\langle S1_n\rangle,\langle f_2^1\rangle))$	$I_1(S)$, $Fl_1(1)$,*	$M(Pc4(\langle S1_n\rangle))\wedge M(PVF(\langle f_2^1\rangle))$	
	$M(Pc4(\langle S2_n\rangle,\langle f_2^2\rangle,\langle f_1^1\rangle,\langle f_2^1\rangle,\langle f_3^1\rangle,\langle f_5^1\rangle))$	$I_2(S)$, $Fl_2(1)$,*	$M(Pc4(\langle S2_n\rangle,\langle f_1^1\rangle,\langle f_2^1\rangle,\langle f_3^1\rangle,\langle f_5^1\rangle))\wedge M(PVF(\langle f_2^2\rangle))$	
	$M(Pc4(\langle S3_n\rangle,\langle f_2^3\rangle,\langle f_1^2\rangle,\langle f_2^2\rangle,\langle f_3^2\rangle,\langle f_5^2\rangle))$	$I_3(S)$, $Fl_3(1)$,*	$M(Pc4(\langle S3_n\rangle,\langle f_1^2\rangle,\langle f_2^2\rangle,\langle f_3^2\rangle,\langle f_5^2\rangle))\wedge M(PVF(\langle f_2^3\rangle))$	
	$M(Pc4(\langle S4_n\rangle,\langle f_2^4\rangle,\langle f_1^3\rangle,\langle f_2^3\rangle,\langle f_3^3\rangle,\langle f_5^3\rangle))$	$I_4(S)$, $Fl_4(1)$,*	$M(Pc4(\langle S4_n\rangle,\langle f_1^3\rangle,\langle f_2^3\rangle,\langle f_3^3\rangle,\langle f_5^3\rangle))\wedge M(PVF(\langle f_2^4\rangle))$	
	$M(Pc4(\langle Sj_n\rangle,\langle f_5^j\rangle))$	$Fl_j(0)$,$Pr_j(1)$,*	$Pc4(\langle Sj_n\rangle))\wedge PVF(\langle f_5^j\rangle))$	(1,2,3,4)
	$M(Pc4(\langle Sj_n\rangle,\langle f_3^j\rangle))$	$Fl_j(0)$,$Pr_j(0)$,*	$M(Pc4(\langle Sj_n\rangle))\wedge M(PVF(\langle f_3^j\rangle))$	(1,2,3,4)
Fr1	$M(PVF(\langle f_1^j\rangle))\wedge M(Pc2(\langle Sj_n\rangle))$	$I_j(N)$,*	$M(Pc2(\langle Sj_n\rangle))\wedge g2(Pc2, f_1^j)\wedge g2(Pc3, f_1^j)\wedge g2(Pc4, f_1^j)$	(1,2,3,4)
	$M(PVF(\langle f_1^j\rangle))\wedge M(Pc3(\langle Sj_n\rangle))$	$I_j(N)$,*	$M(Pc3(\langle Sj_n\rangle))\wedge g2(Pc2, f_1^j)\wedge g2(Pc3, f_1^j)\wedge g2(Pc4, f_1^j)$	(2,3,4)
	$M(PVF(\langle f_1^j\rangle))\wedge M(Pc4(\langle Sj_n\rangle))$	$I_j(N)$,*	$M(Pc4(\langle Sj_n\rangle))\wedge g2(Pc2, f_1^j)\wedge g2(Pc3, f_1^j)\wedge g2(Pc4, f_1^j)$	(1,2,3,4)
Fr3	$M(PVF(\langle f_3^j\rangle))\wedge M(Pc2(\langle Sj_n\rangle))$	$Pr_j\ (1)$,*	$M(Pc2(\langle Sj_n\rangle))\wedge g2(Pc2, f_3^j)\wedge g2(Pc3, f_3^j)\wedge g2(Pc4, f_3^j)$	(1,2,3,4)
	$M(PVF(\langle f_3^j\rangle))\wedge M(Pc3(\langle Sj_n\rangle))$	$Pr_j\ (1)$,*	$M(Pc3(\langle Sj_n\rangle))\wedge g2(Pc2, f_3^j)\wedge g2(Pc3, f_3^j)\wedge g2(Pc4, f_3^j)$	(2,3,4)
	$M(PVF(\langle f_3^j\rangle))\wedge M(Pc4(\langle Sj_n\rangle))$	$Pr_j\ (1)$,*	$M(Pc4(\langle Sj_n\rangle))\wedge g2(Pc2, f_1^j)\wedge g2(Pc3, f_1^j)\wedge g2(Pc4, f_1^j)$	(1,2,3,4)
Fr4	$M(PVF(\langle f_4^j\rangle))\wedge M(Pc2(\langle Sj_n\rangle))$	$Fl_j(0)$,*	$M(Pc2(\langle Sj_n\rangle))\wedge g2(Pc2, f_4^j)\wedge g2(Pc3, f_4^j)$	(1,2,3,4)
	$M(PVF(\langle f_4^j\rangle))\wedge M(Pc3(\langle Sj_n\rangle))$	$Fl_j(0)$,*	$M(Pc3(\langle Sj_n\rangle))\wedge g2(Pc2, f_4^j)\wedge g2(Pc3, f_4^j)$	(2,3,4)
Fr5	$M(PVF(\langle f_5^j\rangle))\wedge M(Pc4(\langle Sj_n\rangle))$	$Fl_j(1)$,$Pr_j(1)$,*	$M(Pc2(\langle Sj_n\rangle))\wedge g2(Pc4, f_5^j)$	(1,2,3,4)

Table 4. Fault and recovery transition definition.

PVF place diagnoses subsystem faults. IF dynamics can also be diagnosed by collecting diagnosed faults temporal information. Table 5 shows the information required for IF dynamics diagnosis:



Fault	*CNT*	*FT*	*T*	T_A	*DF*	*Ps*	*LST*	*OTR*

Table 5. Fault information.

Where *CNT, FT, T,* T_A*, DF, Ps, LST,* and *OTR* have the same meaning as in equations (1), (2), (3), (4), (5), and (6).

The diagnoser builds a table like table 5 for each IF. Moreover, the diagnoser updates the tables each sampling time. Figure 13 shows the updating algorithm.

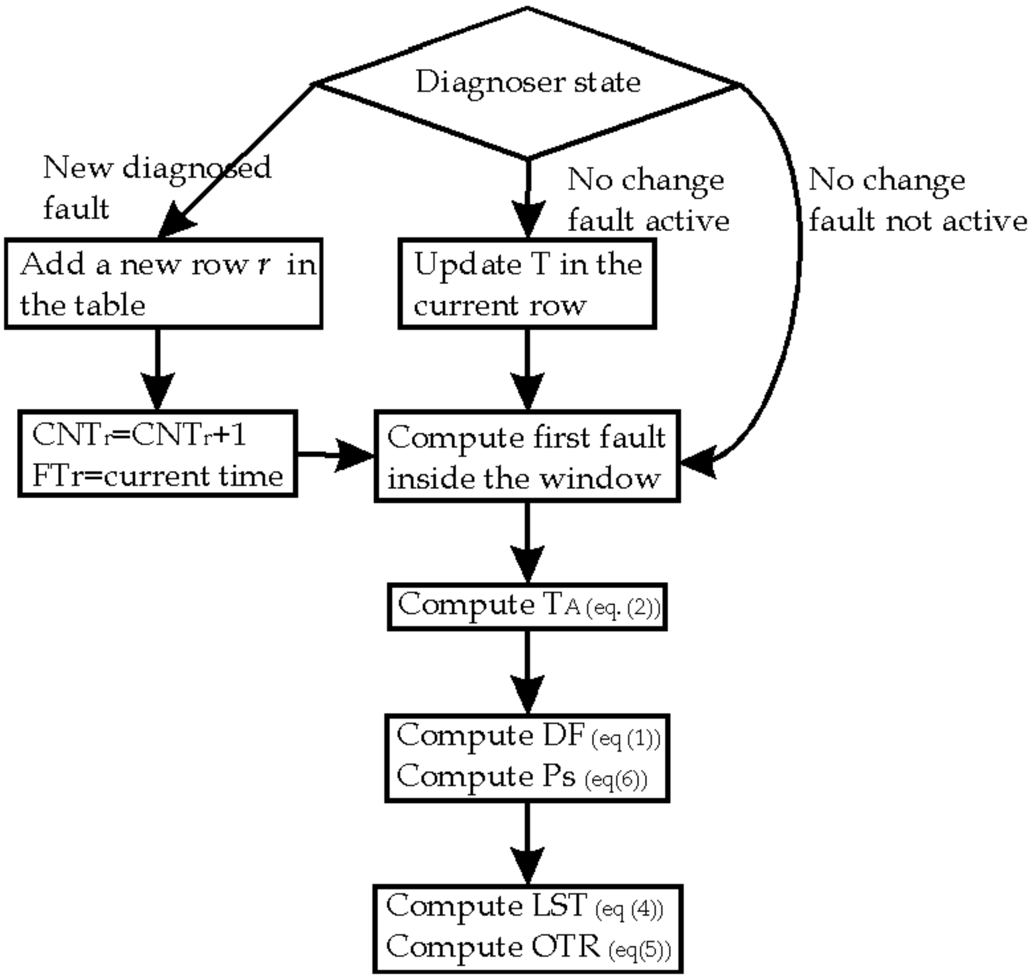


Fig. 13. Updating algorithm.

Therefore, the diagnoser computes *LST* and *OTR* each sampling time. Moreover, each table must be cleared when the faulty device is replaced with a new one.

5. Conclusions

This chapter presents the problem of diagnosing IFs dynamics. We have presented a way of modelling IF dynamics and we have tested it with real data. The density model allows us to compute the best time to repair or substitute the faulty device. This model does not need historical data.

IF dynamics diagnosis has the problem of determine a sliding window size. This problem will be treated in a future work.

We have stated a new definition for IF dynamics and we have proved that the model is able to diagnose the IF dynamics.

We have also presented the integration of IF dynamics diagnosis within a diagnosis technique for discrete event systems based on CPNs. This integration allows the acquisition of temporal information required to compute density and pseudoperiod. Therefore, the diagnosis system will be able to diagnose the IF dynamics.

6. References

Cassandrass, C.G. & Lafortune, S. (1999) *Introduction to Discrete Event Systems.* Kluwer academic publishers.

Contant, O.; Lafortune, S. & Teneketzis, D. (2004) "Diagnosis of intermittent failures".
Discrete event dynamic systems: Theory and applications. Vol. 14. pp. 171-202. Correcher,
A.; García, E.; Morant, F.; Quiles, E. & Blasco. R. (2004) "Intermittent Failure
Diagnosis based on discrete event models". *Proceeding of 7'Th Workshop On Discrete Event Systems WODES04.* pp 151-157. David, R. & Alla, H. (2005) "*Discrete,
Continuous, and Hybrid Petri Nets*", Springer-Verlag, Berlin.

García., E.; Rodríguez., L.; Morant., F.; Correcher., A. & Quiles., E. (2008) "Latent Nestling Method: A new fault diagnosis methodology for complex systems" *Proceeding of The 34th Annual Conference of the IEEE Industrial Electronics Society, IECON08.*

Hensen, K. (1997). "*Coloured Petri Nets Basic Concepts, Analysis Methods and Practical Use*", Volume 1, Second Edition, Springer Editions.

Jensen, F. (2003). *Electronic component reliability.* John Wiley and Sons Ltd.

Jiang, S.; Kumar, R. & Garcia, H.E. (2003) "Diagnosis of repeated/intermittent failures in Discrete Event Systems". *IEEE transactions on robotics and automation.* Vol. 19. n°2. pp 310-323.

Leemis, L. M. (1995). *Reliability: Probabilistic Models and Statistical Methods.* Prentice Hall.

Rodríguez., L.; García., E.; Morant., F.; Correcher., A. & Quiles, E. (2008). "Application of Latent Nestling Method using Coloured Petri Nets for the Fault Diagnosis in the Wind Turbine Subsets". *Proceedings of ETFA'08,* Hamburg, Germany, 2008

Smolensky, P.; Mozer, M. & D. Rumelhart, D. *Mathematical perspectives on neural networks.* Mahwah, NJ: Lawrence Erlbaum Publishers. 1996

Sorensen, B.A.; Kelly, G.; Sajecki, A. & Sorensen, P.W. (1998) "An analyzer for detecting aging faults in electronic devices". *Proceeding of AUTOTESTCON '94. IEEE Systems Readiness Technology Conference and Proceeding of 'Cost Effective Support Into the Next Century'.* Page(s): 417 -421. Available: http://www.usynaptics.com