An approach for diagnosability analysis and sensor placement for continuous processes based on evolutionary algorithms and analytical redundancy

Rubén Leal, Jose Aguilar, Louise Travé-Massuyès, Edgar Camargo, Addison Ríos

▶ **To cite this version:**

Rubén Leal, Jose Aguilar, Louise Travé-Massuyès, Edgar Camargo, Addison Ríos. An approach for diagnosability analysis and sensor placement for continuous processes based on evolutionary algorithms and analytical redundancy. International Journal of Applied Mathematical Sciences, 2015, 9 (43), pp.2125 - 2146. ⟨10.12988/ams.2015.52122⟩. ⟨hal-01701569⟩

HAL Id: hal-01701569

https://hal.science/hal-01701569v1

Submitted on 21 Nov 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

HAL Authorization

Applied Mathematical Sciences, Vol. 9, 2015, no. 43, 2125 - 2146
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/ams.2015.52122

An Approach for Diagnosability Analysis and Sensor Placement for Continuous Processes Based on Evolutionary Algorithms and Analytical Redundancy

Rubén Leal [a], José Aguilar [b, d], Louise Travé-Massuyès [c]
Edgar Camargo [a] and Addison Ríos [b]

[a] Automation Department, PDVSA La Salina 4013, Av. Hollywood
Cabimas, Venezuela

[b] CEMISID, Universidad de los Andes, Av. Alberto Carnevalli
La Hechicera 5101, Merida, Venezuela

[c] DISCO Department, LAAS-CNRS, 7 avenue du Colonel-Roche
31077 Toulouse, France

[d] Researcher Prometeo, Universidad Técnica Particular de Loja, Loja, Ecuador

Copyright © 2015 Rubén Leal et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract

In this work we propose to use an approach based on genetic algorithms to obtain analytical redundancy relations to study the diagnosability property on a given continuous system, and if this not fulfill, our approach allows studying the sensor placement problem in order to fulfill it. The redundancy relations are based on the minimal test equation support and in a structural analysis over a bipartite graph. The faults analysis is studied using a multi-objective fitness function in two genetic algorithms which describe the different constraints to be covered in order to reach the diagnosability property on the system. Additionally, our approach allows studying the sensors placement problem on systems that do not fulfill the detectability or isolability properties, using another genetic algorithm.

Keywords: Genetic Algorithm, Diagnosability, Structural Analysis, Bipartite Graph, Sensors Placement

1. Introduction

Basically, fault means any change in the behavior of any of the components of the system, so that it cannot longer fulfill the function for which it was designed [1]. A diagnosis system consists in the detection and isolation of a set of faults. A diagnostic utilizes observations, i.e. measurements from the system under studied, to determine if a specific behavioral mode is present in the system or not.

In general, the diagnosis analysis has been studied in the literature from two points of view: the Fault Detection and Isolation (FDI) community, which bases the foundations of its solution approaches on engineering disciplines such as control theory and statistical decision making; and the Diagnosis community (DX), which bases the foundations of its solution approaches on the fields of computer science and artificial intelligence. Each community has developed its own terminology, tools, techniques, and approaches to solve diagnosis problems [2]. Our approach is a mixed between the two communities' theories, using a mathematical model and intelligent techniques to solve the problems.

Particularly, when a system has the diagnosability property it can detect and isolate all considered faults. It is a very important property that a given system should meet. The problem of the analysis of diagnosability in the area of continuous processes is very hard, and there are not a lot of works [3, 4]. Another problem linked to this one is the sensor placement problem, which consists in to determine where place the sensors to reach the diagnosability property on the system. There are several works to solve this problem [5, 6, 7], but there are not works which study both problems together.

This paper addresses these problems together. We propose an approach for the analysis of diagnosability, based on a hybrid approach between structural models and intelligent techniques. For that, we identify the different structural models of diagnosis for a diagnosability analysis in the area of continuous processes. Additionally, we use a Genetic Algorithm (GA) to find the set of analytical redundancy relations. In this way, in this paper we propose an analysis of diagnosability based on the residual generation, applying GA to find a set of redundancy relations. In addition, we also propose a sensors placement for fulfill diagnosability using an evolutionary approach. In this case, different combinations of sensors placement are studied with the GA in order to find the most advantageous in terms of diagnosability.

This paper is organized in the following way. The next section presents some concepts about the fault diagnosis problem. Section three presents our hybrid approach of analysis of diagnosability and localization of sensors for continuous processes. Finally, the last section presents some experiments and analyzes the results.

2. Structural analysis based on residuals for diagnosability

In a fault diagnosis system there are three key concepts: The *fault detectability* is the ability to detect certain faults. The diagnosis must be able to decide if there is a fault or not, as well as determine the instant of the apparition, from observations of the process. For that, it needs to compare the current behavior with the expected behavior of the system. The *fault isolation* is the ability to isolate a fault that has occurred, from the faults that are detectable. The *fault identification* studies the differences between the normal behavior and the current behavior in the system, in order to determine the depth and magnitude of the faults.

The fundamental concept in our work is the diagnosability. We start with its definition and the definition of the others close concepts.

Definition 1: Diagnosability. A system has this property if it can detect and isolate all considered faults [4, 8]. The faults that are isolate in a process are often referred as monitored faults, whereas the faults that not isolate are called non-monitored faults.

A typical approach for diagnosability in dynamic systems is the model based diagnosis (MBD) [9]. In the MBD, the fundamental aspects are the definition of a process model, the comparison of the functioning of the process with the model, and the analysis of the behavior of them. A possible MBD technique is based on the generation of residue [9].

There are several technics for the generation of residuals, but all consist in measurements. This paper is based on the analytical redundancy relations approach (ARR) [2]. It uses analytical mathematical models that characterize the system, to reproduce the behavior of the components of the system under studied. The approach for generating ARRs is a finite sequence of calculations that ends with the definitions of analytically redundant equations, which only contains measured or known variables, and is composed of a subset of equations from the model. A residue is a signal ideally zero in the non-faulty case and non-zero else.

The residue generation approaches have in common that the systems should be over-determined [2]. Several algorithms for calculating ARR from over-determined systems have been proposed [8, 10].

2.1. Structural analysis based on residuals for diagnosability.

The structural analysis is a set of tools to explore the fundamental properties of a system using a structural model, either in the form of a bipartite graph or incidence matrix [3]. In our work, a structural analysis of the system is used for the faults detection and isolation, following the approaches used by the community of Fault Detection and Isolation (FDI) [1].

Previous works has modeled the diagnosability based on FDI approaches for continuous processes [9, 11]. At the following we present a resume about the theoretical aspects used by our approach.

2.2 Structural model

A structural model is a representation of a system in which only couplings between variables and equations are retained [12]. The structural model contains only the information of which variable belongs to which equation, regardless of the value of the parameters and the detailed form of the mathematical expression [8]. The structural model can be represented in different way: like bipartite graph, incidence matrix, etc.

Consider a model *M (X,Y,E,F)*, where *E* is a set of equations, $E = \{e_1,..., e_m\}$, *X* is a set of unknown variables, $X=\{x_1,...,x_n\}$, *Y* is a set of known variables, $Y=\{y_1,...,y_p, u_1,...,u_g\}$, and *F* is a set of faults on the system, $F=\{f_1,...,f_o\}$, all considered as known variables. Additionally, $Z = X \cup Y$ is a set of variables of the system. In the case where there are differential or integral variables in the model, it is necessary a fourth set, $D = \{\dot{x}_1, \dots, \dot{x}_n\}$, which contains the derivatives of the variables of X.

Example 1: consider

$$\begin{aligned} e_1&: \dot{x}_1 = -x_1 + u + f_1 \\ e_2&: \dot{x}_2 = x_1 - 2x_2 + x_3 + f_2 \\ e_3&: \dot{x}_3 = x_2 - 3x_3 \\ e_4&: y_1 = x_2 + f_3 \\ e_5&: y_2 = x_2 + f_4 \\ e_6&: y_3 = x_3 + f_5 \end{aligned} \tag{1}$$

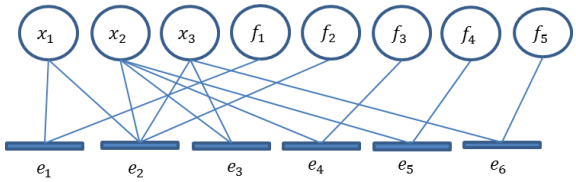

Fig. 1. Bipartite Graphs of the example.

This structure of the system is a representation as it is shown in Figure 1, with the variables involved in the different equations. This abstraction allows us to study the diagnosability properties, independently of the linear or nonlinear nature of the systems. However, it must be kept in mind that results obtained with such a structural representation are a best case scenario. Now, we present some definitions to use the structural analysis for diagnosis purposes.

Definition 2: ARR for M (X,Y,E,F). Let *M(X,Y,E,F)* be a model, then an equation $r_i: h(y, \dot{y}, \ddot{y}, \dots) = 0$ is an ARR for *M(E,X,Z,F)*,if for each *y* consistent with *M(E,X,Y,F)*, the equation is fulfilled [2].

These relationships can be derived only if the model has more equations than unknown variables, i.e. if the system is structurally over-determined (SO) [13].

Example 2: According to example 1, an ARR would be:

$$r_1 = y_1 - y_2 = f_3 - f_4 \quad (2)$$

It is determines by the substitution of all unknown variables by known variables in a single analytical equation. For example, with the equations 4 and 5 we obtain r_1. In this example, the residual r_1 is influenced by the faults $\{f_3, f_4\}$. An ARR can be used to check if the observed variables z are consistent with $M\ (E,X,Z,F)$, and can be used as the basis of a residual generator.

Each r_i should be evaluated in order to decide if it can be used or not. Finally, the evaluation of each detection test constitutes the fault signature vector ($S = \{S_1, \ldots, S_n\}$), that is a set of vectors in order to isolate the fault, where each S_i is the set of residues that are activated by a failure.

Given a set of vector ($S = \{S_1, \ldots, S_n\}$), and a set of faults $F=\{f_1, \ldots, f_o\}$ the theoretical fault signature matrix can be defined codifying the effect of every fault in a residual [12].

Definition 3: *The fault signature matrix of M*. It is a table obtained by the concatenation of all possible signatures of faults. Each row corresponds to an *ARR* and each column to a failure mode. A "1" in position (*ij*), indicates that the fault *j* is detected by the *ARR i* [1].

F (M) is the set of faults that affect either equation in *M*, then the diagnosability ability is achieved if the system complies with the following definitions.

Definition 4: Detectability for M (E,X,Z,F). A fault F_o, where $o = 1,\ldots,n$ which belongs to *F(M)* in the diagnosis system of *M*, is detectable if there is a residue different from zero in the residual generator, i.e $r_i \neq 0$.

Definition 5: Isolability for M (*E,X,Y,F*). When two signatures are identical, the related faults are considered non-decoupled, that mean they cannot be isolated [14]. Therefore, all signatures must be different from each other $S(f_o) \neq S(f_t), \forall o,t \in \{1,\ldots,n\}$, $o \neq t$. The fault isolation will consist in looking for the theoretical fault signature in the fault matrix that matches with the observed signature, to distinguish between all the possible faults.

Example 3: Consider a diagnosis system containing a set of residuals $\{Arr_1, Arr_2, Arr_3, Arr_4, Arr_5, Arr_6\}$, constructed to detect and isolate five faults $\{f_1, f_2, f_3, f_4, f_5\}$. The following fault signature matrix shows the sensitivity of ARRs to faults even in the system in normal behaviour (*N*). Arr_1 is sensitive to faults f_1, f_2, and f_5 and so on. Additionally $S(f_1) = \{Arr_1, Arr_2, Arr_4\}$, as is shown in Table 1.

Table 1. Fault signature matrix

Arr	Signature Matrix					
	N	f_1	f_2	f_3	f_4	f_5
Arr_1	0	1	1	0	0	1
Arr_2	0	1	1	0	1	0
Arr_3	0	0	0	0	1	1
Arr_4	0	1	0	1	0	1

We adopt the design method of minimal structurally over determined (MSO) sets based on ARR like in [10], where provides an algorithm that identifies the MSO, enabling the construction of more efficient ARRs. Each ARR correspond to an MSO.

In [8] introduced an algorithm and the notion of TES (Test Equation Support) which are sets of equations which express redundancy specific to a set of considered faults. Each TES corresponds to a set of faults which influence the residual generator constructed from the TES. The corresponding quantities expressing minimal redundancies are denoted minimal TES (MTES), and the set of MTES can be seen as a subset of the set of MSOs for the set of faults of interest of the system.

Since there is a one-to-one correspondence between MTESs and ARR, we will only focus on MTES's in this paper, to generate residues of a process based on the FDI approach, in order to study the diagnosability property in a system [11]. With that, we will be able build a signature matrix.

Many practical problems are modeled like the interaction between two different types of objects, i.e. between equations and variables, and can be expressed like a bipartite graph problem. For that we give the following definitions.

Definition 6: Bipartite Graph. is an ordered triple $G=(Z,E,\Gamma)$ such that Z and E are sets of vertices, $Z \cap E = 0$, $Z = Y \cup X \cup F$, and Γ are the set of arcs in G.

Definition 7: Matching. is a set of edges from graph G, where each arc has a node from Z an E.

These definitions will help us solve the problem defined in the previous section of search of MTES's, and thus ARR. All the definitions made in this part will be used to apply intelligent techniques in the optimizing of search of paths in a bipartite graph, unlike [4, 8].

3. Our algorithm of diagnosability

In a previous work we implement GAs for the redundancy analysis [11]. This paper attempts to incorporate another GA to reach the full criteria of diagnosability when it is not the case on the system [11]. For that, our new GA solves the sensors allocation problem on the system. Previous GA-based approaches for the sensors placement problem [5, 6, 7] has been proposed. But our algorithm solves both problems together, the analyses of the diagnosability property and then, if it is not fulfill, it solves the sensors placement problem to reach it.

3.1 General Algorithm

In order to fulfill this criterion we developed three algorithms which during their executions invoke several GAs. The first algorithm (called the *MAIN*) simply calls the others two algorithms (*Detection* and *Placement*). The aim of the Detection algorithm is to check whether all the faults in study are detected, (see lines 1-2, of the main algorithm). In the line 2, if the detectability property is not satisfied (MTES (f) = 0 $\forall$ f =1, #*faults*), or the isolability property is not satisfied ($ind\_opt \neq 0$), where $ind\_opt \neq 0$ is a control variable of the second algorithm), the third algorithm (*Placement*) is invoked in order to assign new sensors in unknown variables to increase system redundancy.

Algorithm 1 **Main(G)**

```
1   MTES=Detection(G);
    % It determines detectability and isolability of
    % the system
2     If MTES(f) = 0 ∀ f = 1,#faults or ind_opt ≠ 0
3       Placement (G);
        % It tries to introduce the diagnosability  % property in  a system.
         % to placing sensors in unknown variables
4     end
5   end
```

The second algorithm (called *Detection*) searches an alternative matching in the model (a matching represent a redundancy), based on the theory of MTES. This algorithm has been developed in [11]. The algorithm finds the possible connections between variables and equations in order to eliminate the unobserved variables to fulfill the specific requirements of an ARR. Particularly; it will find populations of MTES invoking the first GA, called GA1. This algorithm also determines the detectability and the isolability (for the last case uses a second GA, called GA2, which uses the best MTES generated by GA1 to build the fault signature matrix).

In the line 2 of the second algorithm is called *GA1* to determine all the possible MTES for each fault (see line1), the lines 4 to 6 guarantee that at least one MTES has been defined for each fault. The line 6 calls *det*, which determines the set of

faults cover by each MTES and includes them in the set *S*. With this information, the algorithm verifies in line 8 if all the faults are covered, using the procedure *verify(S)*. In this way, it determines the detectability of the model (all the faults can be detected). Then, in line 9 it constructs the fault signature matrix (for that, it calls the procedure *iso*). In order to verify the isolability of the system, the procedure calls a second GA (line 10), called *GA2*, with the fault signature matrix as parameter. If the system is isolable, the best individual of GA2 fulfills the isolability propriety (the system has the diagnosability property). If this property cannot be reached, then it defines $ind\_opt \neq 0$ in order to call in the main algorithm the third algorithm, called *Placemen*t, which is used when there are faults not detected or isolated.

Algorithm 2 **Detection (G)**

1 **for** $k = 1$, $k = \#faults \mid f \in F$
2 $MTES\ (f_k) = GA1\ (f_k, G)$;
% It determines the MTES's for each fault
3 **end**
4 **if** at least $\exists\ MTES\ (f) \neq 0 \mid \forall\ f \in F$
5 **for** $i = 1$, $i = \#\ faults \mid f \in F$;
6 $S(i) = \det\ (i, \text{MTES}\ (f_i))$;
% It determines the set of faults cover by % the MTES's of each fault
7 **end**
8 **if** verify $(S) = True$
% It determines the detectability of the model
9 $S_m = \text{iso}(f_i, S)$;
% It constructs the fault signature matrix
10 $ind\_opt = GA2(S_m)$;
% It determines the isolability of the faults % ($ind\_opt = 0$)
11 **end**
12 **end**

The placement algorithm uses another GA, called GA3, to find a set of possible sensors placement (each individual is a possible sensor placement) based on the failures to detect (unknown variable) and the cost of the sensors, in order to choose the lowest value to save money at the time of implantation (see line 5). The new bipartite graph proposed by GA3, that represents the structural model of the system, is *G´*(see lines 1 and 5). If GA3 converges due to the number of generation, then means that there are not individuals (*G´*) that reach the diagnosilability property (see next section where is explained GA3), that is, this property cannot be reached by the real system (see line 3). Else, the individual G' is the new bipartite graph of the system, and the placement of the new sensors has a cost according to the fitness function of this individual (line 5).

Algorithm 3 **Placement(G)**
% propose some sensors for unknown variables
1 *cost = GA3* (*G,G′, cost*)
2 **If** *GA3* has converged due to Nb_generation **then**
3 **print "**This system has not the diagnosability property"
4 **else**
5 **print** "The cost of the sensors is" cost, "and the configuration is" *G′*
6 **End**

3.2 The GAs in our diagnosability analysis approach

GA is one of the most powerful heuristics tools for solving optimization problems, based on natural selection and the process of biological evolution [5]. A GA repeatedly modifies a population of individual solutions. At each step, the GA selects individuals randomly from the current population to be parents, and uses them to produce the descent for the next generation. Over successive generations, the population "evolves" towards an optimal solution. GAs can be applied to solve a variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, non-differentiable, stochastic, or highly nonlinear. In the next subsection we are going to describe our GAs.

3.2.1 GA1

Given a model *M*, its diagnosability property can be analyzed using a structural model of the system as bipartite graph (*G*), and is represented by the union of all the faults and all the unknown variables $G = G_X \cup G_F$. From G_F, the fault f_k is extracted, GA1 will generate a population of individuals with random combinations of active arcs in the model (an active arc is when the relation between this equation and the variable is selected to compose the possible MTES defined by this individual for this fault f_k, see below for more details). That is, each individual is going to define a possible MTES. The objective function measures if (possible MTES) reaches the detectability proprieties. The best individuals will be the input of the GA2, in order to study the isolability property.

An individual in GA1 is represented by a bits-string, where each bit is used to represent the arcs connecting the vertices in the bipartite graph, a_{ij}, *a* is the arc between vertices, *i* (the unknown variables *x* and *f* in the bipartite graph) and *j* (the equation *e* in the bipartite graph). If an arc is selected (active arc) $a_{ij} = 1$, else $a_{ij} = 0$ (it represents that this arc does not exist in the bipartite graph or is not an active arc). For our example in the section 2, an individual is shown in Figure 2.

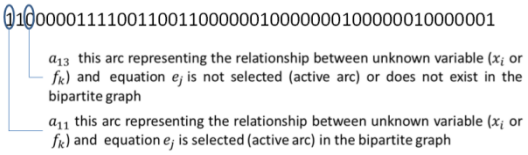

Fig. 2. Bits string representation of an the individual in GA1

Hence, for every possible MTES for this fault *f* there is a unique bits-string sequence. The MTES (individual) is evaluated using fitness function which includes all the important design criteria of a MTES. The general algorithm of GA1is:

```
Genetic Algorithm 1 f_k
1  begin GA1
2      pop=generate_pop(G);
       % Generate initial population
3      Evaluation(FF1, pop);
       % compute the fitness function to each
       %  individual
4      while not FF1 = 0 or Nb_generation do
5          begin
           % reproductive cycle begins
6              Reproduction (pop)
7              Evaluation (FF1, pop)
8              selection (pop)
9          end
10     end
11 end
```

The details of this algorithm are in [11]. This evolutionary process is stopped when is reached a number of generations, or when an individual reaches a value of $FF_1 = 0$ (this MTES represents a residue of the fault f_k).

FF_1 is a multi-objective function in order to find the MTES for the fault *f*. A weighted sum approach has been used to aggregate all the optimization criteria according to the equation (4), gives a numerical value of the quality of each possible solution (MTES) of the optimization problem.

$$FF_1 = Min\left\{\sum_{n=1}^{3} w_n P_n\right\} \tag{3}$$

Where, w_n is the corresponding weight and P_n is the criterion to be optimized. The different criteria to be optimized are in Table 2.

Table 2. Optimization criteria

Objective	Optimization criteria
$\boldsymbol{P_1}$	**Number of Failures**
$\boldsymbol{P_2}$	**Studied Fault**
$\boldsymbol{P_3}$	**System Redundancy**

P_1 ensures that only is studied one fault f_k at a given time: It is described by the sum of the active arcs of the faults that are in the individual, different of the failure to study. If this sum is different from zero is penalized the equation 3 with a weight w_1.

$$P_1 = \sum_{\substack{f_i \in F \\ f_i \neq f_k}} \sum_{j=1}^{m} a_{kj} \tag{4}$$

Where:
a_{kj} are the arcs between the faults and the equations
f_k Select fault
m Number of equation

P_2 ensures that the fault studied f_k is considered by the individual: it will make a sum of the active arcs from the studied fault presents in the individual. Normally, only one active arc from the studied fault must exist to guarantee a TES, otherwise this value is different from zero and the individual must be penalized by a weig.t w_2.

$$P_2 = \left| \sum_{j=1}^{m} a_{f_k j} - 1 \right| \tag{5}$$

where
a_{fkj} are the arcs from the studied fault

P_3 verifies that the degree of redundancy of the model is equal to one, that is, there is one equation more than unknown variables. For that, it checks the cardinality of the active arcs for the case of the variables and equations, and this difference must be one to ensure the propriety of MSO and MTES. This function must also shed as a result the value of zero, otherwise the individual must penalized by weight w_3.

$$P_3 = \left| \left(Card(J^*) - Card(I^*) \right) - 1 \right|_{\substack{I^* = \{i \mid \exists a_{ij} = 1 \ \forall \ j = 1, m\} \\ J^* = \{j \mid \exists a_{ij} = 1 \ \forall \ i = 1, n\}}} \tag{6}$$

where
I^* are the active arcs in unknowns variables
J^* are the active arcs in equations

Having described the design criteria we formalize our fitness function as:

$$FF_1 = w_1 P_1 + w_2 P_2 + w_3 P_3 \tag{7}$$

In this fitness function the importance of each criterion is defined by the weight

w_n. The values are determined according to the design requirements and experimentation (see table 3).

Table 3. Weighting coefficients

Objective	Optimization criteria	Optimized Value
w_1	Number of Failures	100
w_2	Failure in study	1000
w_3	System Redundancy	10000

Particularly, in GA1 three elite individuals (individuals with the best fitness values) of each generation are chosen in order to ensure that the current best individuals always survived in the next generation.

Theorem 1: GA1 ensures that if there is an individual with $FF_1 = 0$ then it is a MTES of f_k , and the set of individuals with $FF_1 = 0$ will be the set of MTES of the studied fault f_k.

Proof of theorem 1; C represents one individual in *G* with $FF_1 = 0$, that is a MTES sensitive a f_i.

3.2.2 GA2

The resulting population from GA1 involves perhaps a large number of MTES. As there is a one to one relationship among the ARRs and MTESs, and if the detectability property is reached in the system, the detection algorithm builds the signature matrix and calls GA2, in order to determine a set of ARRs that guaranteeing the isolability property of the system, if there exist.

The individual in the GA2 is also represented by a bits string. In this case, each bit represents each MTES (ARR) found by GA1 for all the faults, GA2 randomly defines individuals to be subsequently evaluated by the objective function, where a bit with value of "1 " means that a MTES is chosen in the fault signature matrix, otherwise its value is "0". GA2 studies the sensitivity of the faults for the MTES's selected in a given individual, and observes if in this individual the isolability property is reached.

The objective function of GA2 ensures that its value is zero because the individual has isolability property. Otherwise, this individual cannot isolate the faults (isolability property). There are two conditions in the objective function.

$$FF_2 = \sum_{i=1}^{\#f} \sum_{k=i+1}^{\#f} O_{ij}^1 + \sum_{i=1}^{\#f} O_i^2 \tag{8}$$

Where O_{ij}^{1} y O_{i}^{2}

$$O_{ik}^{1} = \begin{cases} 0 & \text{if } \forall j = 1, \#ARRs \ \exists V_{ij} \neq V_{kj} \\ 10 & else \end{cases} \quad (9)$$

$$O_{i}^{2} = \begin{cases} 0 & \sum_{j=1}^{\#ARRs} V_{ik} \neq \emptyset \\ 10000 & else \end{cases} \quad (10)$$

The first condition O^1ensures that the fault signature vector f_i in the matrix constructed by the individual is different than the fault signature vector f_j, otherwise it penalizes with a low value, interpreting than f_i cannot be isolated of f_j. O^2 condition ensures that the sum of each vector in the fault signature matrix is non-zero, otherwise it cannot distinguish the behavior of a system with fault of a normal behavior, penalizing the individual with a high value.

Genetic Algorithm 2 (S_m)

```
1  begin GA2
2     pop=generate_pop (Sm);
      % Generate initial population chosen some   % ARR from (Sm)
3     Evaluation (FF2, pop);
      % compute the fitness function to each
      % individual
4     while not FF2 = 0 or Nb_generation do
5        begin
         % reproductive cycle begins
6           Reproduction(pop)
7           Evaluation (FF2, pop)
8           selection (pop)
9        end
10    end
11 end
```

This GA2 generates the initial individuals using S_m, and the rest of the procedure is a classical GA. In general, each individual selects a set of MTES (e.g. $MTES_1$, $MTES_2$, $MTES_4$ and $MTES_6$), and the GA2 will check the signatures of each fault in the set of MTES selected using the objective function FF_2.

Theorem 2: GA2 ensures that if there is an individual with $FF_2 = 0$, it will get a set of MTES which accomplish with the isolability propriety.

Proof of Theorem 2: For a full diagnosability, the value of FF_2 must be equal to 0, all values more than 0 means that at least one fault is not isolable from other faults, and all the values more than 10000 means that there is a signature equal to the model in normal behavior, making not detectable the fault.

3.2.3 GA3

In the case that the system does not fulfill with the diagnosability property (detectability or isolability properties), GA3 proposes a new bipartite graph G'. For that, it randomly selects unknown variables and assumes them as known variables of the system. These unknowns variables suppose as known variable, involve place sensors in the system to make them known (can be measured). The graphs G' are randomly generated (each G' is an individual, see line 3, and the *random_generation(G)* algorithm modifies the graph G including new arcs).

GA2 checks if with that new sensors configuration proposed by each individual then it meets the detectability and isolability properties (for that, GA3 call Detection (G') algorithm, see line 6). Otherwise, it regenerates new G' during a given number of generations. If the GA3 achieve several individuals with this property, then choose the individual with the lowest cost of implementation of its sensors configuration (see line 16), which is determined based on the costs of implantation of the sensors defined in each individual (see line 14).

GA3 is a new algorithm for the sensor placement problem, and its result will be the individual with the lowest cost to reach the diagnosability property.

Genetic Algorithm 3 (G, G´, cost)

1 **begin** AG3
2 **repeat until** size of the population
3 $G' = random\_generation\ (G)$
% propose some sensors for unknown
% variables
4 $pob = pob + G'$
5 **end**
6 **while not** (Detect **(**G'_i **)** $\neq true\ \forall\ G'_i \in pob$ or Nb_generation **do**
7 **begin**
% reproductive cycle begins
8 Reproduction (pob)
9 **end**
10 **if** ∃ Detec(G'_i)$= true\ \forall\ G'_i \in pob$ **then**
11 **begin**
12 **for** $\forall\ G'_i \in pob$ **where** Detect **(**G'_i **)** = *true*
13 **begin**
14 $cost_i = \sum_j^{\#vd}(ind.Cost) * (ind.value)$ % for each sensor there is a cost. Each
% value of 1 on the gen of an individual
% means a sensor
15 **end**
16 cost = min_i ($cost_i$);
% select the cheaper new configuration of
% sensors
17 $G' = G'_i$ where min_i ($cost_i$);
18 **end**

19 **else**
20 *cost* = 0
21 **end**

The individuals for GA3 are structurally similar to the individuals of GA1. The algorithm that randomly generates the population of individuals *G´* carries out a variation of *G.*

Algorithm **random_generation(G)**
1 **for** *k # VD*
% it will place a sensor in a possible un
% known variable
% with a sensor for each one that has a
% cost
2 **Modify** (G)
% according to the new known variables
3 **end**

Theorem 2. GA3 solves the problem of sensor location by randomly proposing a modification of *G* (*G´*), which represents a new structural model of the system, in order to fulfill the property of diagnosability.

Proof of Theorem 3; GA3 solves the sensors placement problem for a system, with a population of new *G´,* in order to reach the diagnosticability propriety.

$$FF_3 = min_i(cost_i) \tag{11}$$

GA3 considers the cost of the new sensors configuration (see equation 11). Thus, GA3 selects the individual with the minimum cost (see line 16) between the individuals with the diagnosability property (see line 10).

4 Experiment

In this section we are going to prove our hybrid approach to solve the diagnosability analysis and the sensors placement problems.

The *Main* algorithm invokes the *Detection* algorithm for the location of a population of sensitive MTES for each fault in the system, in order to reach its detectability property. In addition, it constructs the failure signature matrix with the minimum set of MTES to meet the isolability property in the system. The *Detectio*n algorithm invokes to GA1 and GA2 respectively, to perform this task. Finally, if these properties are not reached, the *Placement* algorithm is invoked to find an assignment of new sensors in the system to reach the diagnosability property.

For the experiment, we will evaluate the example that we have been using through this paper and structurally defined in section 2.

4.2 Detection Problem

The initial population of GA1 is composed by fifty individuals. These individuals are sub-models of the main structural model, which is sufficient to fulfill the initial requirements for MTES. These individuals are evaluated using the fitness function FF_1, and in each generation the two best individuals are selected as parents, to execute the crossover operator with probability 0.7, and the mutation operator with probability 0.9. The stopping criterion is: if the individual with the best fitness value does not improve in 50 iterations AG1 stops. Figure 3 is a representation of an individual $G = G_X \cup G_F$.

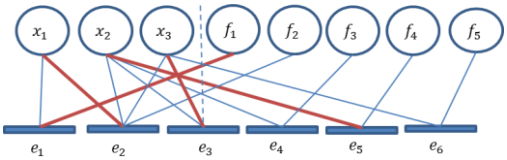

Fig. 3. A possible individual of GA1

Graphically, the figure 3 shows an individual (C_1) in which the objective function FF_1 can be evaluated.

C1;0100000000100010001000000000000000000000000000000

This individual is defined by the red lines shown in the figure 3, (they are the active arcs between the vertices of equations and the vertices of the variables). In the string of C_1 is represented by a value "1". If we evaluate equations 4-6.

$$P_1 = \sum_{\substack{f_i \in F \\ f_i \neq f_1}} \sum_{j=1}^{6} a_{kj} = 0 \tag{12}$$

P_1 means that there is only one active arc from the set of vertices belong to F.

$$P_2 = \left| \sum_{j=1}^{6} 1 - 1 \right| = 0 \tag{13}$$

P_2 ensures that there is only one active arc from the studied fault. System redundancy is evaluated by P_3, where in the individual C_1 there are 4 equations and 3 variables with active arcs.

$$P_3 = |(\ Card(3) - Card(4)) - 1| = 0 \tag{14}$$

$$FF_1 = 100*(0)+1000*(0)+10000*(0)=0 \tag{15}$$

This proof theorem 1, that C_1 represents one individual in G with $FF_1 = 0$, and it is a MTES sensitive to f_1.

4.3 Isolation

Continuing with the same example, we will evaluate the fault signature matrix showed in the table 1, to proof the theorem 2 (isolability of the model).

GA2 starts with a population of 20 individuals evaluated by the objective function FF_2. Then, like GA1, the best individuals are chosen as parents in each generation to apply the crossover and mutation operators. The process to generate new individuals is performed carefully, because each individual must contain valid arcs among the possible arcs of the original bipartite graph. The stop criterion in this case is when the best individual does not improve after 20 iterations.

As we said before, an individual in GA2 is represented by a bit string. For this experiment we will evaluate two individuals A_i, where 1 in an individual means that this ARR in the signature matrix is chosen.

A_1: 011100000
A_2: 001011010

For example, in the case of A_1 are selected Arr_2, Arr_3, Arr_4. In Table 4 we will show the fitness function of each individual for the first condition. It will compare the signature of each fault.

The best result en O^1 is "0" that is when an individual has the isolability property. In this case none of the individual fulfill with this proprieties because the signature of f_1 and f_2 are similar. In Table 5 is studied the second condition.

Table 4. Evaluation of O^1 of the individuals chosen.

O_1	Evaluation of the fitness function									
	f_1 f_2	f_1 f_3	f_1 f_4	f_1 f_5	f_2 f_3	f_2 f_4	f_2 f_5	f_3 f_4	f_3 f_5	f_4 f_5
A_1	10	10	0	0	0	0	0	0	0	10
A_2	10	0	0	0	0	0	0	0	0	0

Table 5. Evaluation of O^2 of the individuals chosen

O^2		Evaluation of the fitness function				
		f_1	f_2	f_3	f_4	f_5
1	A_1	0	0	0	0	0
2	A_2	10000	10000	0	0	0

We can see that the sum of the values for A_1 is equal to 0, means that those individual are different of the model of the normal behavior. With respect to A_2, it has the same model of the normal behavior, which means the value is different to zero. Below is the result of the evaluation of the fitness function of GA2.

$$FF_2(A_1) = 30 + 0 = 30$$
$$FF_2(A_2) = 10 + 20000 = 10010$$

According to theorem 2, for a full diagnosability, the value of FF_2 must be equal to 0. All values more than 0 means that at least one fault is not isolable from other faults, and all the values more than 10000 means that there is a signature equal a the model in normal behavior, making no detectable the fault. Table 6 shows the final signature matrix proposed by GA2.

Table 6. Result of GA2

Arr	Signature Matrix					
	N	f_1	f_2	f_3	f_4	f_5
Arr_1	0	1	1	0	0	1
Arr_2	0	1	1	0	1	0
Arr_3	0	0	0	0	1	1
Arr_4	0	0	0	1	0	1

4.4 Sensor placement problem

In the case that the model does not fulfill with the diagnosability, the main algorithm calls the placement algorithm to solve the placement problem. This algorithm randomly generates new G' to search a sensors configuration which will reach the diagnosability propriety. Placement algorithm calls GA3 algorithm in order to evaluate G'.

As we said before, an individual in GA3 is like an individual in GA1. For this experiment, we will evaluate three individuals which are modification of the original G. For example, in Figure 4 is shown a possible G' placing a sensor in x_1, adding a new equation e_7: $y_4 = x_1$.

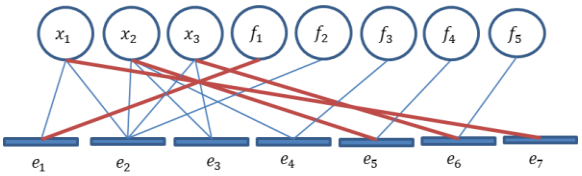

Fig. 4. A possible G' placing a sensor in x_1

For this individual from GA3, we define several individuals (a population) for GA1. For example, C_1' is one of them (it is composed by the active arcs of the Figure 4, red lines).

In Table 7, it is shown the evaluation of C_1' in GA1, and the signature matrix placing a sensor in x_1 is shown in the Table 8, as the result of GA2, with the specific ARRs:

Table 7. Evaluation of G' in GA1

Populations		Evaluation of the fitness function				
		P_1	P_2	P_3	P_4	Value of FF_1
1	C_1'	0	0	0	0	0

Table 8. Result of GA3 Placing a new sensor x_1

Arr	Signature Matrix Placing a new sensor x_1					
	N	f_1	f_2	f_3	f_4	f_5
Arr_1	0	1	1	0	0	1
Arr_2	0	1	1	0	1	0
Arr_3	0	0	0	0	1	1
Arr_4	0	0	0	1	0	1

GA3 solves the sensors placement problem for a system, with a population of new G', in order to reach the diagnosability propriety.

Figure 5 shows another possible individual of GA3 (it is called G_2') placing a sensor in x_3 (e_7: $y_4 = x_3$). The signature matrix placing a sensor in x_3 is shown in the Table 9.

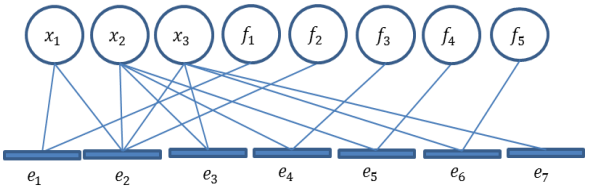

Fig. 5. Another possible individual of GA3, called G_2'

Similarly, we need to call GA1 and GA2. Particularly, evaluating the best individual we can see that does not fully comply with the isolability property because f_1 and f_2 cannot be isolated from each other, which is determined by the evaluation of GA2 in Table 10.

In this way, GA3 can discriminate among good individuals or not, in order to propose a new configuration of sensors to reach the diagnosability property on the system.

Table 9. Result of GA3 placing a sensor in x_3

Arr	Signature Matrix Placing a new sensor x_3					
	N	f_1	f_2	f_3	f_4	f_5
Arr_1	0	1	1	0	0	0
Arr_2	0	0	0	1	0	1
Arr_3	0	0	0	1	1	1
Arr_4	0	0	0	0	1	1

5 Conclusions

The capability to detect a fault on time in a system provides security, availability and reliability. The fault diagnosis mechanisms used in this paper is based on the principles of redundancy. This paper analyzed the technique of structural analysis, in order to propose a hybrid approach for the diagnosability and sensor placement problems for continuous processes. Our hybrid approach is based on a structural model and a GA.

Specifically, we use several GAs to solve our problem. A first GA has a population of bipartite graphs, the result is the set of MTES to detect all the faults in the system. The second GA allows reach the isolability property, selecting the minimum number of MTES in the fault signature matrix. The last one solves the sensors placement problem.

Our approach is an efficient tool to solve the combinatorial problem behind of the determination of MTESs, given a structural model as a bipartite graph. Additionally, it determines where new sensors must be placed (again, it is a combinatorial problem), in order to reach the diagnosability property. That means, our approach carries out several studies in an automatic way: determination of the detectability and isolability of the faults of a system (diagnosability property), and according to the results, assignment of sensors to reach this property. We have shown that GA's is a powerful tool for providing the answers to these questions. The execution time of the GAs can be controlled with their parameters, in order to obtain good solutions in a short computing time.

Particularly, our GA1 studies the detectability property based on the structural analysis of the studied system. It is a hard optimization problem, because there are a lot of alternative paths in a system, which increase according to its number of variables and equations. These paths must be evaluated to determine the detectability property using our FF1 (it carries out a structural analysis in each path). The GA2 studies the isolability property analyzing the fault signature matrix, in order to determine the MTES necessary to reach it. This process of selection of MTES is another optimization problem solves by GA2. If these properties are not reached, the diagnosability property is not reached. In order to reach this property, GA3 studies the problem of placement of sensors. Again, this is an optimization problem because the number and combination of possible new sensors depend on the unobserved variables. These different optimization problems are very well solved

with ours GAs; additionally, theirs execution times can be controlled through theirs parameters (number of generations, numbers of individuals, etc.).

A future work is implement our approach in real continuous processes, like a production system of extraction of oil.

Acknowledgments. This work has been supported by Postgraduate Cooperation Program (PCP) between Venezuela and France entitled "Supervision and maintenance task in a shared organizational environment", between the University of Los Andes, Venezuela with the cooperation of LAAS- CNRS, University of Toulouse, France. Dr Aguilar has been partially supported by the Prometeo Project of the Ministry of Higher Education, Science, Technology and Innovation of the Republic of Ecuador.

References

[1] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, Diagnosis and fault tolerant control. Berlin: Springer, 2003.
http://dx.doi.org/10.1007/978-3-662-05344-7

[2] J. Armengol, A. Bregon, T. Escobet, E. Gelso, M. Krysander, M. Nyberg, X. Olive, B. Pulido, and L. Travé-Massuyès, Minimal structurally overdetermined sets for residual generation, A comparison of alternative approaches, In Proceedings of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, pages 1480–1485, 2009.
http://dx.doi.org/10.3182/20090630-4-es-2003.00241

[3] L. Travé-Massuyès, T. Escobet, and X. Olive. Diagnosability analysis based on component supported analytical redundancy. IEEE Trans. on Systems, Man, and Cybernetics – Part A: Systems and Humans, 36(6): pages 1146–1160, 2006.
http://dx.doi.org/10.1109/tsmca.2006.878984

[4] C. Svard and M. Nyberg. Residual generators for fault diagnosis using computation sequences with mixed causality applied to automotive systems, IEEE Transactions on Systems, Man, and Cybernetics – Part A, 40(6): pages 1310–1328, 2010. http://dx.doi.org/10.1109/tsmca.2010.2049993

[5] S. Sen, S. Narasimhan, K. Deb. Sensor network design of linear processes using genetic algorithms, Comput. Chem. Eng. 22 (3), pp. 385–390, 1998.
http://dx.doi.org/10.1016/s0098-1354(97)00242-1

[6] S. Jin, M. Zhou, A.S. Wu. Sensor network optimization using a genetic algorithm, in: 7th World Multiconference on Systemics, Cybernetics, 2003.

[7] A. Bhondekar, R. Vig, M. LalSingla, C. Ghanshyam, P. Kapur. Genetic Algorithm Based Node Placement Methodology for Wireless Sensor Networks, Pro-

ceedings of the International Multi Conference of Engineers and Computer Scientists 2009 Vol I IMECS, 2009.

[8] M. Krysander, J. Åslund, and E. Frisk, A structural algorithm for finding testable sub-models and multiple fault isolability analysis, In Proceedings of the 21st International Workshop on Principles of Diagnosis (DX-10), 2010.

[9] L. Travé-Massuyès, T. Escobet, and R. Milne. Model-based diagnosability and sensor placement. Application to a frame 6 gas turbine subsystem. In DX01 twelfth international workshop on principles of diagnosis, pages 205–212, 2001.

[10] M. Krysander, J. Åslund, and M. Nyberg, An efficient algorithm for finding minimal over-constrained subsystems for model-based diagnosis, IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans, 38(1), pages 197-206, 2008. http://dx.doi.org/10.1109/tsmca.2007.909555

[11] R. Leal, J. Aguilar, L. Trave-Massuyes, A. Ríos, E. Camargo."A Genetic Algorithm Approach for Diagnosability Analysis", International Journal of Engineering Development and Research, Vol.2, No. 4, pp. 3786-3799, 2014.

[12] M. Krysander, M. Nyberg. Structural analysis utilizing MSS sets with application to a paper plant. Proceedings of the Thirteenth International Workshop on Principles of Diagnosis, 2002.

[13] M. Nyberg. Automatic design of diagnosis systems with application to an automotive engine. Proceedings Control Engineering Practice, 7(8): pages 993–1005, 1999. http://dx.doi.org/10.1016/s0967-0661(99)00054-4

[14] V. Puig, J. Quevedo, T. Escobet, B. Pulido. On the integration of detection and isolation in model based fault diagnosis. Proceeding In DX04, 15th International Workshop on Principles of Diagnosis, 2004.

Received: February 25, 2015; Published: March 20, 2015