AUTOMATIC DESIGN OF DETECTION TESTS IN COMPLEX DYNAMIC SYSTEMS

Stéphane Ploix, Matthieu Désinde, Samir Touaf

Laboratoire d'Automatique de Grenoble, INPG, UJF, UMR 5528,
BP 46, F-38402 Saint Martin d'Hères Cedex, France
Phone: 33 4 76 82 62 44, Fax: 33 4 76 82 63 88

Abstract: In complex industrial plants, there are usually lots of sensors and the modelling of the plant leads to lots of mathematical relations. Before using classical tools for fault detection, the first problem to solve is: what sensors and mathematical relations have to be selected for the design of a detection test such as a state observer or a parity space based detection algorithms. This paper presents a general method for automatically selecting relevant sensors and relations that may be used for the design of the different detection tests. This method, which is based on a structural analysis of the process, provides all the testable subsystems and permits the selection of the most interesting detection tests regarding detectability and diagnosticabillity criteria.

Keywords: Dynamic Systems, Fault detection, Fault diagnosis, Detection algorithms design

1. INTRODUCTION

In the scientific literature, there are two main streams for the design of detection tests. The first one, mostly used by researchers coming from the Fault Detection and Isolation community (Patton et al., 1989), relies on global models of systems to be diagnosed. It is often called structured or robust approach because it aims at projecting residuals in different spaces in order to discriminate the different faults that may occur. Another stream comes from the Artificial Intelligence community (Reiter, 1987). It relies on component based approaches. The principle is to model the different components of the system and to combine these models in order to perform detection tests. The FDI approach mainly deals with dynamic systems whereas the DX approach mainly focus on static system. Detailed studies about the comparison between these approaches have been achieved by the IMALAIA French research group (Cordier et al., 2000) and by the BRIDGE action of the European network of excellence called MONET (http://monet.aber.ac.uk: 8080/monet).

The MAGIC European project (Köppen-Seliger et al., 2002) has shown that bridge approaches between the 2 communities (Nyberg and Krysander, 2003; Ploix et al., 2003), taking advantage both of detection tools for dynamic system and of formal reasoning for fault isolation are very suitable for industrial plants (Garcia-Beltram et al., 2003). These bridge approaches are, on one hand component based, i.e. each component state is individually modelled, and in the other hand, they cope with sophisticated detection tests. If the systems are simple, the design of detection tests may be easily handled but if the system becomes a little bit more complex, this task becomes unachievable. For instance, a bioprocess modelled by 34 mathematical relations has led to 736 possible detection tests. It can obviously not be solved by hand.

A detection test has to be distinguished from its support, called Testable Sub System (TSS) in (Ploix et al, 2003). A TSS gathers the constraints used for its design and the hypotheses the test relies on. A given set of constraints may be combined in many different ways and, providing that the set is testable, leads to many different detection tests but all of them

are checking the same part of the system to be diagnosed: they are all based on the same TSS. Designing a fault detection procedure may be decomposed into two steps: identifying the parts of the system to be diagnosed that can be tested (the TSS) and, in a second step, choosing for each TSS the most relevant algorithm for designing a detection test (state observer or parity relation based algorithm, for instance). Discovering TSS can be achieved thanks to a procedure based on a structural model such as the bi-partite graph approach (Dulmage and Mendelsohn, 1959) proposed in (Straroswiecki and Declerck, 1989) (Declerck and Staroswiecki, 1991) and in (Blanke et al, 2003). Finding testable subsets may indeed be done thanks to an elimination procedure that combines constraints related to each component in order to eliminate all the unknown physical variables and therefore getting constraint containing only known data (i.e. testable constraints). In (Boutobza, 2003), it is shown that this approach does not provide all the possible testable subsystems: the more sensors there are, the less analytical redundancy relations are found.

This paper present a new algorithm based on elimination rules, which improves the algorithm proposed in (Ploix and Follot, 2001). It relies also on a structural analysis of the constraints. Comparing to algorithms based on Gröbner bases (Frisk, 2000), this algorithm may be used whatever the nature of the constraints is; nonlinear differential equations can for instance be easily handled. Moreover, the proposed algorithm traces all the elementary constraints which are involved in a final testable constraint. This is a very important characteristic because it leads to the support of each detection test, which is required in formal diagnostic analyses. However, a drawback of this algorithm is that it may lead to unachievable testable subsystems, which have to be removed afterwards, especially if calculability has not been taken into account from the beginning.

2. DIRECTED STRUCTURAL MATRIX

Let's describe the notation used. Observations stand for directly observable facts or events, which contain information about an actual physical state of the system to be diagnosed. Physical variables represent the phenomena according the Kantian definition. They have to be distinguished from model parameters, which cannot be directly observed because they depend on a model. Voltage and current are physical variables, resistance value not. Physical variables are therefore model independent. These variables will be putted in square brackets. For instance, voltage and current will be denoted as $[v]$ and $[i]$. Known values related to physical variables are also denoted in a special way: they are topped by a "~" sign. For instance, a known value for a voltage $[v]$ will be denoted $\tilde{v}$. Known values are generally observations coming either from control variables or from measurements, but they sometimes may be known by assumption: consider for instance a surrounding temperature that may be assumed to be between $20°^c$ and $25°^c$. Known value is a key notion in fault diagnosis because they contain the available information about actual system states.

Each component can be modelled by a constraint, corresponding to a relation between physical variables. Each constraint C can be structurally abstracted over a set of physical variables V containing all the physical variables appearing in the constraint, by a couple C containing two parts:

- the structure $\iota_V(C)$ of a constraint, which is a $\dim(V)$-dimensional vector. Each element may be 0, 1 or -1. The null value at i^{th} position means that the i^{th} physical variable does not appear in C. A unitary value means that the i^{th} physical variable appears in C and that it may be deduced from the other physical variables of C. The value -1 means that the i^{th} physical variable appears in C but that it cannot be deduced from the other physical variables of C due to some invertibility problems.
- the set of references $\sigma(C)$, also called the support of the constraint, containing either the name of the constraint if it is related to only one component state, or, elsewhere, the names of all the constraints that compose C.

Consider for instance the following constraint:

$$\left([z] = f_1([x],[y]),\{C\}\right)$$

and the set of physical variables $V=\{[x],[y],[z]\}$.

If it is possible to calculate these functions:

$$\begin{cases}[y] = f_2([x],[z]) \\ [x] = f_3([y],[z])\end{cases}$$

then, this constraint is without causality (in the sense of calculability) (Iwasaki and Simo, 1994). Because each variable may be deduced from the others, the structure of the constraint will be written:

$$\iota_V(C)=[1, 1, 1]$$

Assume now that the function f_3 cannot be obtained. In this case, the variable $[x]$ cannot be deduced from the others. Therefore, the structure of the constraint will be written:

$$\iota_V(C)=[-1, 1, 1]$$

Definition: Two constraints C_1 and C_2 defined on a set of physical variables V, will be considered as equivalent if $\iota_V(C_1)= \iota_V(C_2)$ and if $\sigma(C_1)=\sigma(C_2)$. It will be denoted $C_1 \Leftrightarrow C_2$.

Definition: A constraint C_2 overestimates a constraint C_1 if the three following conditions are satisfied:

- $|\iota_V(C_1)| = |\iota_V(C_2)|$ (vectors are equals in absolute value)
- $\iota_V(C_2)-\iota_V(C_1) \geq 0$ (each element of the difference has to be positive or null)
- $\sigma(C_1) \subset \sigma(C_2)$

It will be denoted $C_1 \subset C_2$.

Constraints overestimating others have to be removed because there are not minimal.

The proposed algorithm aims at searching for all the possible testable subsystems. It is based on a *directed structural matrix* of the system define on a set of physical variables V. This matrix summarizes the physical variables (but not the parameters) involved in the constraints of the system to diagnose. The rows of the matrix are related to the constraints: each row corresponds to a constraint C_i. It is worth the structure of the related constraint $\iota_V(C_i)$.

Information about calculability may be forgotten i.e. '1' can be putted instead of '-1'. Nevertheless, if that's how it is, the algorithm will lead to unachievable detection tests, which will have to be removed afterwards. It is indeed considered that other variables cannot be deduced from the remaining ones because the use of the derivative of the state variable has to be avoided. Note that, contrary to the requirements of the digraph approach, variables appearing in the derivative do not have to be modelled with a new virtual variable.

The directed structural matrix is then cleaned up in order to improve the convergence towards the solutions i.e. the set of all the possible testable subsystems. The following algorithm is applied:

```
Do until no longer elimination is possible
- remove empty columns of the directed
  structural matrix
- remove columns containing only one '1' and
  no '-1'
```

Then, the set of constraints S appearing in the directed structural matrix and the involved physical variables are sorted so that the structural matrix becomes upper block triangular (1). The resulting matrix is called re-arranged directed structural matrix. Constraints are arranged so that each block located on the diagonal is split into 2 sub-blocks denoted $T_i(S) \times V_i(S)$ and $P_i(S) \times V_i(S)$.

$$M = \begin{bmatrix} \ddots & \vdots & \vdots & \vdots \\ & P_2(S)\times V_2(S) & P_2(S)\times V_1(S) & P_2(S)\times V_0(S) \\ & T_2(S)\times V_2(S) & T_2(S)\times V_1(S) & T_2(S)\times V_0(S) \\ & & P_1(S)\times V_1(S) & P_1(S)\times V_0(S) \\ & & T_1(S)\times V_1(S) & T_1(S)\times V_0(S) \\ & & & P_0(S)\times V_0(S) \\ & & & T_0(S)\times V_0(S) \end{bmatrix} \quad (1)$$

Constraints belonging to $T_0(S)$ contain only one physical variable. These constraints are described as 0-order terminal in S. It is usually composed by behavioural constraints modelling sensors or actuators, which link one physical variable to one known value. Physical variables intervening in $T_0(S)$ form the set $V_0(S)$ namely 0-order physical variable in S. $P_0(S)$ are constraints containing only physical variables in $V_0(S)$ and that do not belong to $T_0(S)$. There are called 0-order packed constraints. These notations may be generalized to any order. The terminal relations of i^{th} order in S, called $T_i(S)$, are relations that contains only one variable belonging to $V_i(S)$ and no variable in $V_j(S)$: $\forall j>i$. If a relation contains several variables in $V_i(S)$ and no variable in $V_j(S)$: $\forall j>i$, it belongs to $P_i(S)$.

Therefore, the order of a constraint is defined with respect to the sets $V_i(S)$ coming from the upper triangular rearrangement of the structural matrix related to S. It corresponds to the maximum order of the physical variables present in the constraint. If this order is equal to i, the constraint C is qualified as i^{th}-order in S and it is denoted $i=\text{order}_S(C)$.

By extension, the order of a set of constraints S is equal to the maximum order of the physical variables present in S. It is denoted $order(S)$.

3. ELIMINATION RULES

Re-arranged directed structural matrix is then used to generate potential testable subsystems according to 2 elimination rules, partially presented in (Ploix and Follot, 2001). These rules may be used to combine 2 constraints C_1 and C_2 with one common variable in order to get the composition of a new constraint, only if $C_1 \not\eqslantless C_2$ and $(C_1 \not\preceq C_2) \vee (C_2 \not\preceq C_1)$. A common variable is chosen to be eliminated if one of the following rules may be applied (of course, the eliminated variable will not be present in the newly generated constraint):

Rule 1/1 may be applied if the common variable to be eliminated may be deduced in both original constraints i.e. two '1' are present in the column of the variable to be eliminated. If at least one of the constraints contains several '1', then every remaining '1' becomes '1' in the new constraint. The variables remaining in the original constraints become '-1'.

Rule 1/-1 may be applied if the common variable to be eliminated may be deduced in only one constraint i.e. one '1' and one '-1' are present in the column of the variable to be eliminated. Then, every '1' presents in the constraint where the common variable cannot be deduced, i.e. on the same row that the eliminated '-1', become '1' in the new constraint. All the other remaining variables become '-1', even if there are '1' in one of the original constraints.

Moreover, for both rules, the support of the new constraint is equal to the union of the support of the original relations. Let C_3 be a new constraint coming from an elimination between two original constraints C_1 and C_2 then, $\sigma(C_3) = \sigma(C_1) \cup \sigma(C_2)$.

These rules can be easily proven in considering that a directed constraint can be written as:

$$[x_0] = f([x_0], [x_1], [x_2], \ldots)$$

(if x_0 is present among inputs, the constraint is a equation to be solved) and in considering that a non-directed constraint satisfies :

$$g([x_0],[x_1],[x_2],...)=0$$
$$\Rightarrow \exists g_0(.)/[x_0]=g_0([x_1],[x_2],...)$$
$$\Rightarrow \exists g_1(.)/[x_1]=g_1([x_0],[x_2],...)$$
$$\Rightarrow \exists g_2(.)/[x_2]=g_2([x_0],[x_1],...)$$
...

Anyway, whatever the applied rule is, the elimination of the k^{th} variable of a set V between 2 constraints C_i and C_j can be denoted by:

$$C_l = r_V(C_i, C_j)$$

As an example, consider the following constraints:

$$\begin{cases} C_1: \ \dfrac{d[h_1]}{dt} = \dfrac{[q_1]-[q_2]}{S_1} \\ C_2: \ [q_2]=\alpha_1[h_1]\theta_2 \end{cases}$$

Because derivation of variables has to be avoided, it leads to the following directed structural matrix:

	q_1	q_2	h_1
C_1	-1	-1	1
C_2	0	1	1
$C_1 \cup C_2$	-1	0	1
$C_1 \cup C_2$	1	1	0

Physical variables q_2 and h_1 are common and may be alternatively eliminated using respectively rule 1/-1 and rule 1/1. Two new constraints of which the support is known ($\{C_1,C_2\}$), may be generated. However, when the number of constraints is important, eliminations have to be organized in order to avoid redundant operations.

4. ELIMINATION WITHIN A SET OF CONSTRAINTS

Before describing the global algorithm, let's define the *block elimination* i.e. elimination within a set of constraints instead of the *one by one elimination* between two constraints.

The elimination within sets of i^{th} order terminal or packed constraints belonging to a set S is denoted:

$$S' = \Pi_{P_i}^{P_i}(S),\ S' = \Pi_{P_i}^{T_i}(S) \text{ or } S' = \Pi_{T_i}^{T_i}(S)$$

where S is the initial constraint set and S' is the resulting constraint set.

This operator generates new constraints resulting from all the possible one by one eliminations between constraints within $T_i(S)$ and $P_i(S)$ that successively eliminate the variables of $V_i(S)$. Block elimination is an iterative procedure. The number of iterations is equal to the number of variables in $V_i(S)$. Assume that card($V_i(S)$)=n with $V_i(S)=\{x_0, ..., x_k, .., x_{n-1}\}$. The block elimination starts by initializing a constraint set $S_0=S$. Then, the first variable x_0 of $V_i(S)$ is considered. After all the possible one by one eliminations of x_0 between constraints in $T_i(S_0)$ and $P_i(S_0)$ (depending on the letters in the Π operator), have been done according to elimination rules, the constraints involved in the eliminations are removed from S_0 and the newly generated constraints are added to S_0 in order to produce the new set S_1 providing that the constraints overestimating or equivalent to other constraints of S_1 are removed. Not removing constraints involved in eliminations lead to a variant of the block elimination called *conservative block elimination.* It is only used between packed constraints:

$$S' = \underline{\Pi}_{P_i}^{P_i}(S)$$

This operation is reiterated with x_1 within $T_i(S_1)$ and/or $P_i(S_1)$ … until S_{n-1} is reached. The solution of the block elimination S' satisfies $S'=S_n$. The initial set of constraints S_0 is thus gradually updated. Because of the eliminations, the following property is necessary satisfied: $order(S') \leq order(S)$. The block elimination ends when it remains no more possible elimination. Consider the following directed structural matrix:

Table 1 – Example of directed structural matrix

S	x_0	x_1	x_2
C_0	0	-1	1
C_1	1	0	1
C_2	-1	0	1
C_3	0	1	1
C_4	0	0	1

According to the definitions, $S=\{C_0,C_1,C_2,C_3,C_4\}$, $V_0(S)=\{x_2\}$, $V_1(S)=\{x_0,x_1\}$, $T_0(S)=\{C_4\}$ and $T_1(S)=\{C_0,C_1,C_2,C_3\}$. As an example, let's calculate: $S' = \Pi_{T_1}^{T_1}(S)$. S_0 is initially set to S.

Firstly, x_0 is considered. It is eliminated between C_1 from $T_1(S_0)$ and C_2 from $T_1(S_0)$ using the 1/-1 elimination rule (C_1 cannot be eliminated with itself). It leads to a new constraint C_5 of which the support is $\{C_1,C_2\}$. C_5 contains only x_2 as a possible output. The elimination of x_0 between C_2 and C_1 is not computed because it is equivalent with the previous elimination. The constraint C_5 is added to S_0 and C_1 and C_2 are removed from S_0 in order to produce the updated sets $S_1=\{C_0,C_3,C_4,C_5\}$. The new constraint C_5 belongs to $T_0(S_1)$.

Thus, Table 1 becomes:

S_1	x_0	x_1	x_2
C_0	0	-1	1
C_3	0	1	1
C_4	0	0	1
C_5	0	0	1

Now, the set $T_1(S_1)$ is composed of $T_1(S_1)=\{C_0,C_3\}$. The variable x_1 is then eliminated between C_0 and C_3 using 1/-1 rule. It leads to constraint C_6, which only contains x_2 as a possible output. The constraint set S_2 becomes:

S_2	x_0	x_1	x_2
C_4	0	0	1
C_5	0	0	1
C_6	0	0	1

The set $T_1(S_2)$ is now empty. Therefore, the block elimination $S' = \Pi_{T_1}^{T_1}(S)$ is finished. Only three constraints remain: one initial constraint (C_4) and two new constraints C_5 and C_6. Consider the set S defined by table 2.

Table 2 – Example of directed structural matrix

S	x_0	x_1	x_2
C_0	0	-1	1
C_1	1	1	1
C_2	-1	0	0
C_3	0	1	0
C_4	0	0	1

According to the definitions, $V_0(S)=\{x_0,x_1,x_2\}$, $T_0(S)=\{C_2,C_3,C_4\}$ and $P_0(S)=\{C_0,C_1\}$. As an example, let's calculate: $S' = \Pi_{P_0}^{T_0}(S)$. The order of S is obviously equal to 0. The initial set is $S_0=S=\{C_0,C_1,C_2,C_3,C_4\}$. The variable x_0 may be eliminated between C_1 from $P_0(S_0)=\{C_0,C_1\}$ and C_2 from $T_0(S_0)$ using the 1/-1 rule. It leads to a new constraint C_5. Because there is no more possible elimination of x_0, the constraint C_1 and C_2 may be removed from S_0 and the constraint C_5 is added to S_0 in order to produce $S_1=\{C_0,C_3,C_4,C_5\}$:

S_1	x_0	x_1	x_2
C_0	0	-1	1
C_3	0	1	0
C_4	0	0	1
C_5	0	1	1

Two new eliminations may then be performed between $P_0(S_1)=\{C_0,C_5\}$ and $T_0(S_1)=\{C_3,C_4\}$. The variable x_1 may be eliminated between the constraint C_0 from $P_0(S_1)$ and C_3 from $T_0(S_1)$, and between C_3 from $T_0(S_1)$ and C_5 from $P_0(S_1)$. It leads respectively to constraints C_6 and C_7. The constraints C_0, C_3 and C_5 are removed from S_1:

S_2	x_0	x_1	x_2
C_4	0	0	1
C_6	0	0	1
C_7	0	0	1

The block elimination $S' = S_2 = \Pi_{P_0}^{T_0}(S)$ is finished because the set $P_0(S_2)$ is now empty. Two new constraints have been generated whose supports are: $\sigma(C_6)=\{C_0,C_3\}$ and $\sigma(C_7)=\{C_1,C_2,C_3\}$.

5. GLOBAL ELIMINATION PROCEDURE

The principle of the global elimination procedure is to progressively reduce the maximum order of the constraint set until only 0-order constraints remain. Then, testable subsystems called *basic testable subsystems* will be obtained by directly eliminating the remaining 0-order variables of the $P_0(S)$ constraints using exclusively $T_0(S)$ constraints. Nevertheless, other testable subsystems may still be found by eliminating physical variables between $P_0(S)$ constraints and afterwards, eliminating the remaining variables using $T_0(S)$ constraints. These testable subsystems are called *complex testable subsystems*. Finally, Testable subsystems may still be found by eliminating variables within $T_0(S)$: some of the generated constraints may be related to material redundancy.

Firstly, the directed structural matrix, composed by all the behavioural constraints of the system, is re-arranged i.e. clean up and sorted as mention at the beginning of section IV. In order to gradually eliminate highest order variables, the following algorithm is used:

```
Set n=order(S),
If P_n(S)≠{∅} and T_n(S)={∅} then
  -S becomes Π_{P_n}^{P_n}(S)
  -remove remaining n-order constraints from
  S
Do from n=order(S) until n=1
  -if P_n(S)≠{∅}, S becomes Π_{P_n}^{T_n}(S)
  - S becomes Π_{T_n}^{T_n}(S)
  -remove remaining n-order constraints from
  S
```

When order 1 is reached, two blocks remains in S: $T_0(S)$ and $P_0(S)$. Last eliminations have now to be performed.

The *basic testable subsystems* are then given by:

$$\Sigma_B = \Pi_{P_0}^{T_0}(S) \oplus \Pi_{T_0}^{T_0}(S)$$

These subsystems are qualified as *basic* because they result from the shortest way of getting testable subsystems. It means that they check the largest possible part of the system with the minimal number of constraints.

Constraints within $P_0(S)$ can also be recombined in order to get more complex testable subsystems. These *complex testable subsystems* are given by:

$$\Sigma_C = \Pi_{P_0}^{T_0}(S') \oplus \Pi_{T_0}^{T_0}(S') \text{ where } S' = \underline{\Pi}_{P_0}^{P_0}(S)$$

This way of eliminating physical variables is systematic but not unique. The elimination can also be done in a random way. Nevertheless, the proposed procedure reduces the number of elimination in order to get all the testable subsystems.

6. APPLICATION EXAMPLE

This algorithm has been applied to an electric circuit (figure 1) depicted by the following constraints:

$$\left\{\begin{array}{|c|c|c|}
\hline
\{[e]-[v_1]=R[i_1],\{C_1\}\} & \left\{C\frac{d[v_1]}{dt}=[i_2],\{C_5\}\right\} & \{\tilde{i}_2=[i_2],\{C_9\}\} \\
\hline
\{[v_1]=[v_2],\{C_2\}\} & \{\tilde{e}=[e],\{C_6\}\} & \{\tilde{i}_3=[i_3],\{C_{10}\}\} \\
\hline
\{[v_1]=[s],\{C_3\}\} & \{\tilde{s}=[s],\{C_7\}\} & \{\tilde{v}_2=[v_2],\{C_{11}\}\} \\
\hline
\{[i_1]=[i_2]+[i_3],\{C_4\}\} & \{\tilde{i}_1=[i_1],\{C_8\}\} & \\
\hline
\end{array}\right.$$

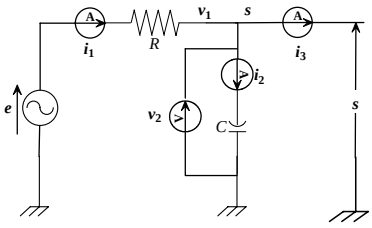

Figure 1 – Electrical circuit

Constraint C_1 is related to the resistor (R), C_2, C_3 and C_4 to a node between electric cables (N), C_5 to the condenser (C), C_6 to a generator (G) and C_7 to C_{11} to sensors. Several questions arise such as what can be tested? What are the diagnosticable components? The proposed algorithm permits to solve these issues.

The elimination procedure leads to the following testable subsystems (the constraints have been replaced by the components), summarized in a signature table:

	(R)	(N)	(C)	(G)	(S)	(I_1)	(I_2)	(I_3)	(V_2)
TSS_1	0	1	0	0	1	0	0	0	1
TSS_2	1	1	0	1	0	1	0	0	1
TSS_3	1	1	0	1	1	1	0	0	0
TSS_4	0	0	1	0	0	0	1	0	1
TSS_5	0	1	0	0	0	1	1	1	0
TSS_6	0	1	1	0	1	0	1	0	0
TSS_7	1	1	1	1	0	1	1	0	0
TSS_8	1	1	1	1	0	0	0	1	1
TSS_9	1	1	1	1	1	0	0	1	0
TSS_{10}	1	1	1	1	1	0	0	1	1
TSS_{11}	1	1	1	0	0	1	0	1	0
TSS_{12}	0	1	1	0	0	1	0	1	1
TSS_{13}	0	1	1	0	1	1	0	1	0
TSS_{14}	1	1	1	1	0	0	1	1	0
TSS_{15}	1	1	0	1	0	0	1	1	1
TSS_{16}	1	1	0	1	1	0	1	1	0

It exists 16 testable subsystems (5 basic). It is of course not necessary to test all of them. But this signature table shows the best performance that can be expected from any diagnostic system. Because the columns are independent, it is possible to infer that all the components are diagnosticable. It is also possible to foresee the performances if only the 5 basic testable subsystems are performed: resistor and generator are indistinguishable but other components remains diagnosticable.

7. CONCLUSION

This paper presents a general method to compute all the testable subsystems of a system to be diagnosed. The proposed elimination procedure has been designed in order to reduce as much as possible the number of computations. The provided testable subsystems are very useful because they lead to signature tables; it permits to foresee the best possible achievable performances of diagnostic procedures. Because the proposed algorithm is based on a structural analysis of the process, it also applies to non linear systems. Overestimation of solutions may be reduced in taking into account the calculability in the constraints. These results are very useful for bridge approaches of fault diagnosis, which require the specification of the support of each detection test.

8. REFERENCE

Blanke, M., M. Kinnaert, J. Lunze and M. Staroswiecki (2003). Diagnosis and Fault-Tolerant Control, Springer Verlag, Heidelberg, Germny.

Boutobza, A. (2003). "Génération automatique des tests de détection pour le diagnostic de systèmes physiques complexes." Laboratoire d'Automatique de Grenoble, Grenoble.

Cordier, M.-O., P. Dague, M. Dumas, F. Lévy, J. Montmain, M. Staroswiecki, and L. Travé-massuyès (2000). A comparative analysis of AI and control theory approaches to model-based diagnosis. *14th European Conference on Artificial Intelligence*, Berlin, Germany.

Declerck, P., and M. Staroswiecki (1991). Characterization of the canonical components of a structural graph for fault detection in large scale industrial plants. *European Control Conference*, Grenoble, France.

Dulmage, A.L. and N.S. Mendelsohn (1959). A structure theory of bi-partite graphs of finite exterior extension, Transaction of Royal Society of Canada, Section III, vol.53, pp. 1-13.

Frisk, E. (2000). Residual Generator Design for Non-linear, Polynomial Systems - A Gröbner Basis Approach. *IFAC Fault Detection, Supervision and Safety for Technical Processes*, Budapest, Hungary.

Garcia-Beltram, C., S. Lesecq, and R. Stücher (2003). Diagnosis of a hydraulic looper and a hot rolling mill main drive in a multiagent framework. *IAR-ICD/IFATIS/MAGIC Workshop on Advanced Control and Diagnosis - ACD 2003*, Duisburg.

Iwasaki, Y., and H. A. Simo (1994). Causality and model abstraction. *Artificial Intelligence*, **67**, pp. 143-194.

Köppen-Seliger, B., S. X. Ding, and P. M. Frank (2002). MAGIC - IFATIS: EC-Research Projects. *15th IFAC World Congress*, Barcelona, Spain.

Nyberg, M., and M. Krysander (2003). Combining AI, FDI, and Statistical Hypothesis-Testing in a Framework for Diagnosis. *IFAC Safeprocess'03*, Washington, U.S.A.

Patton, R., P. Frank, and R. Clark (Eds) (1989). *Fault diagnosis in dynamic systems*, Prentice Hall.

Ploix, S., and C. Follot (2001). Fault diagnosis reasoning for set-membership approaches and application. *CCA/ISIC'01*, Mexico City, Mexico.

Ploix, S., S. Touaf, and J.-M. Flaus (2003). A logical framework for isolation in fault diagnosis. *SAFEPROCESS'2003*, Washington D.C., U.S.A.

Reiter, R. (1987). A theory of diagnosis from first principles. *Artificial Intelligence*, **32**, pp. 57-95.

Staroswiecki, M. and P. Declerck (1989). Analytical redundancy in nonlinear interconnected systems by means of structural analysis, IFAC AIPAC'89 Symposium, Nantes, France.

9. Acknowledgement

This work is supported by the European Commission under the project EU-IST-2000-30009 MAGIC