Diagnosing photovoltaic systems: an iterative process

Abed Alrahim Yassine, Stéphane Ploix, Hussein Joumaa

G-SCOP lab
46, avenue Félix Viallet F-38031 Grenoble cedex 01

Abed-Alrahim.Yassine@g-scop.inpg.fr, Stephane.Ploix@inpg.fr, Hussein.Joumaa@imag.fr

***Abstract*— This paper presents the principles of a fault diagnosis system for photovoltaic plants that makes it possible iterative diagnosis processes. It proposes a solution for auto generating tests from models and consecutive measurements. Moreover, it is shown that overall possible diagnoses may be compute from the symptoms coming from the generated tests. The operation principles are detailed and technological aspects for implementation are discussed. Thanks to these diagnosis systems, maintenance operators just have to depict the photovoltaic plant and to introduce iteratively the measurements they do. The system to be diagnosed is transformed into a MILP Problem (Mixed Integer Linear Programming) and a GLPK solver is used to find out all the possible diagnoses.**

***Index Terms*— Fault diagnosis, detection test design, Symptom generation, constraint satisfaction problem, MILP solver.**

I. Introduction

Today's competitive global economy requires continued advances in fault diagnostic technology to achieve and maintain cost advantages. This challenge can be envisaged by using, as much as possible, efficient diagnostic systems that makes it possible to detect an isolate the faults the wich occurs in systems.

In this paper, the model-based diagnosis method has been adopted because of its analysis capabilities and its generality. This method provides an alternative to the traditional techniques based on experience, such as rule-based or case-based reasoning systems. In the model-based method, tree main kinds of approaches might be found: The FDI approach, which focuses on fault detection in dynamic systems, has been summarized in [2]. The community **FDI** (Fault Detection and Isolation ([19], [12], [4], [10], [14]) bases the foundations of its solving approaches on engineering disciplines such as control theory and static decision making. The DX approach, summarized in [13], focuses on the diagnosis reasoning. The main works of the **DX** approach ([22], [7]) base the foundations of solution approaches on the field of computer science and artificial intelligence. Recently, a bridge approach between FDI and DX has been proposed ([17], [20]).

The bridge approach has been adopted for the design of diagnosis system. It can be decomposed into two stages: detection and isolation. In the first stage, the detection tests are computed for the system to be diagnosed from the system model composed of a set of constraints. Then, symptoms are deduced from these detection tests. One of the proposed methods for the symptom generation is to transforme the system to be diagnosed into a MILP Problem (Mixed Integer Linear Programming) which is a regular constraint satisfaction problem and to use GLPK solver which makes it possible to compte symptomes.
In the second stage, the fault diagnosis is performed from the generated symptoms.

In order to find out the detection tests, several sensors have to be installed. The performance of a diagnostic system highly depends on the number and on the location of actuators and sensors. Several sensor placement methods are presented in this paper.
[23] has proposed a method based on consecutive retraction of sensors, which takes into account diagnosability criteria. This approach is based on Analytical Redundancy Relations (**ARR**) [2]. However, this method requires an a priori design of all the **ARR** for a given set of sensors.
Recently, the sensor placement problem satisfying diagnosability objectives becomes possible without designing **ARR**s. [11] has proposed a efficient method based on a Dulmage-Mendelsohn decomposition [9]. Nevertheless, this method only applies to just-determined sets of constraints while most practical systems are under-determined when sensors are not taken into account, and over-determined afterwards.
Another sensor placement method without designing **ARR**s has be presented in ([27], [28]). This method improves the possibility of detecting and localizing faults in systems for which only the structure is known. It considers the complete range of specifications with respect to the constraints, i.e. the set of constraints that must be diagnosable, the set of constraints that must be non discriminable but detectable and the set of constraints that must be non detectable.

The paper is organized as follows. Section II describes the necessary steps for fault diagnosis. Section III presents the concepts used to model systems in a standardized form for diagnosis. Section IV details the methods to find out all the testable subsystems. Section V presents a new method of symptom generation from the testable subsystems by transforming the system to be diagnosed into a MILP Problem and by using a GLPK solver. In section VI, the diagnosis reasoning method is presented. In section VIII, the steps of the fault diagnosis system is applied on a photovoltaic

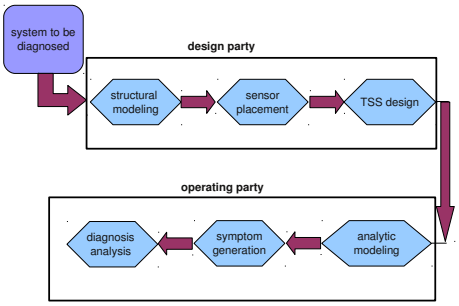

Fig. 1. Fault diagnosis system

system. Finally, the conclusions are drawn in Section IX.

II. Fault diagnosis system schema

The goal of a fault diagnosis system is to detect and isolate the possible faults which can occur in systems. The necessary steps of the diagnostic system are given in figure 1.

1) the first step is the structural modeling of the system to be diagnosed. This step consists in colleting a structural model of the systsem behavior
2) the second step is the sensor placement. This step makes it possible to find the optimal sensor placement satisfying objectives like observability, monitorability and diagnosability
3) the third step is the design of testable subsystems (**TSS**s) i.e a constraint set which leads to a test. This step makes it possible to compute the necessary **TSS**s for diagnosis
4) the fourth step is the analytical modeling of the system to be diagnosed. This step consists in colleting a model the system behavior according to section III
5) The fifth step is the symptom generation. This step makes it possible to generate the symptoms from the **TSS**s found.
6) The last step is the diagnostic analysis. This step computes out all the diagnoses from the symptoms found.

III. System modeling for diagnosis

This section deals with concepts used to model systems in a standardized form for diagnosis. Let us firstly introduce some basic definitions.

A variable models an information, material or energy flow. It is named *shared variable* when it is shared by several components. Shared variables play the role of *port* ([6]) for connection between components. Variables are therefore potentially observable elements of information about the actual state of system. It is important to distinguish between a variable which is related to a physical phenomena from a parameter which is model-dependent. Generally speaking, even if the variable is observable, it is not possible to merge it with data because in fault diagnosis, data are only known providing that some actuators or sensors behave properly. This fact leads to the concept of *data flow*. It is used to model data provided by a source of information concerning a variable. This source may be a sensor or an actuator. The set of all possible data flows, that can be collected from a system when spanning all the possible values for controlled variables, is named OBS. A set of observed values is written obs.

In fault diagnosis, a system is not supposed to remain in a given mode. Indeed, diagnostic analysis aims at retrieving the actual behavior modes of the components of a system. At least, two modes have to be defined: the *ok* mode, which corresponds to the normal behavior and *cfm* mode, which is the complementary fault mode: it refers to all the behaviors that do not fit to the normal behavior. Sometimes, specific fault modes may be modeled but in this paper, for the sake of clarity, only the modes ok and cfm are considered.

Except for the complementary fault mode, behavioral modes are modeled by cause-effect relationships between variables. These relationships are represented by constraints. Each constraint refers to a set of mappings containing unknown variables and known data flows represented by $obs \in OBS$. This fact can be formalized as follows:

$$ok(c_i) \leftrightarrow \forall obs \in OBS, k_i(obs, V_i) = 0 \quad (1)$$

where $k_i(obs, V_i) = 0$ is the constraint set to be satisfied by component c_i in mode $ok(c_i)$ and V_i is the set of variables involved in k_i.

However, it is obvious that in most of practical fault diagnosis problems, it is not possible to check all the context. Equation (1) leads to:

$$ok(c_i) \rightarrow \forall obs \in \mathcal{O}, k_i(obs, V_i) = 0 \quad (2)$$

where $\mathcal{O} \subset OBS$.

For the sake of simplicity, elementary models are formalized by:

$$(ok(c_i), k_i(obs, V_i) = 0) \quad (3)$$

Consequently, the model of a system Σ is composed of a set of components and a set of behavioral modes related to these components.

Let's now present the way of constructing the system model. A complex system can be broken into a set of connected components. In other words, a system can also be constructed by joining together its components. The model of a component is assumed to be embedded into it thanks to a communication mechanism: component could be named *smart* or *intelligent components*. This model can represent one of the possible modes (*ok* or *cfm* here). As mentioned before, a component can connect others via its variables. Several components can be joined together via a connection point, which is a shared variable. The connection points between the different components of a system have to be specified by the installer. A variable can also be connected to sensors which provide data flows.

In order to illustrate the fault diagnosis system, consider for instance the photovoltaic system shown in figure 2.

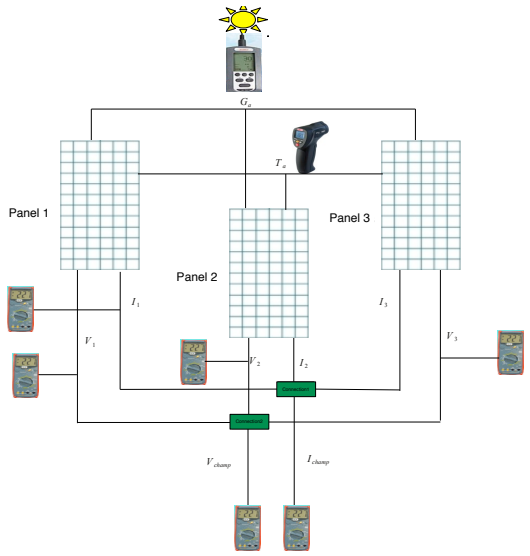

Fig. 2. A photovoltaic system composed of 3 panels.

This system is basically composed of tree main panels Photowatt PW1650 each: $panel_1$, $panel_2$ and $panel_3$. The panels are connected in series. The variables of $panel_1$ are ambient irradiation, denoted $panel_1.G_a$, the ambient temperature denoted $panel_1.T_a$, the cell temperature denoted $panel_1.T_c$, the panel current denoted $panel_1.I$, the panel voltage denoted $panel_1.V$, the short circuit current $panel_1.I_{sc}$ and the open circuit voltage denoted $panel_1.V_{oc}$. The model corresponding to the normal behavior (the *ok* mode) of $panel_1$ is given by the constraints k_{1-1}, k_{1-2}, k_{1-3}, k_{1-4} and k_{1-5}. In these constraints, the mentionned variables of $panel_1$ are indicated, for short, by: G_a, T_a, T_c, I_1, V_1, I_{sc_1} and V_{oc_1}.

$$\begin{array}{l}(ok(panel_1), k_{1-1} : I_{sc_1} = 5.3 \times G_a)\\ (ok(panel_1), k_{1-2} : T_c = T_a + 0.031 \times G_a)\\ (ok(panel_1), k_{1-3} : V_{oc_1} = 0.6 - 0.0023(T_c - 25))\\ (ok(panel_1), k_{1-4} : V_{t_1} = 0.000345(273 + T_c))\\ (ok(panel_1), k_{1-5} : I_1 = \ldots\\ I_{sc_1}(1 - exp((V_1 - 72 \times V_{oc_1} + 0.45 \times I_1)/72V_{t_1})))\end{array} \quad (4)$$

By replacing the constraints k_{1-1}, k_{1-2}, k_{1-3}, k_{1-4} in the constraint k_{1-5}, the following constraint is obtained:

$$\begin{array}{l}k_1 : I_1 = 5.3 \times G_a(1 - exp((V_1 - 72(0.6 - \ldots\\ 0.0023(T_a + 0.031 \times G_a - 25)) + 0.45 \times I_1) \ldots\\ /72(0.000345(273 + T_a + 0.031 \times G_a)))).\end{array} \quad (5)$$

For a given measure of G_a, the photocurrent can be calculated by the following relationship:
$I_{ph} = \frac{G}{G_{STC}}[I_{ph,STC} + \alpha(T_c + T_{STC})]$ where:
G: irradiation received by the photovoltaic cell
T_c: the cell temperature
α: coefficient of temperature of the short circuit current
G_{STC}: the irradiation at standard condition $[1000W/m^2]$
T_{STC}: the temperature at standard condition $[25^C]$
$I_{ph,STC}$: the photocurrent at standard condition

In the same, The model corresponding to the normal behavior (the *ok* mode) of $panel_2$ and $panel_3$ is given by the constraints k_2 and k_3 .

$$\begin{array}{l}k_2 : I_2 = 5.3 \times G_a(1 - exp((V_2 - 72(0.6 - \ldots\\ 0.0023(T_a + 0.031 \times G_a - 25)) + 0.45 \times I_2) \ldots\\ /72(0.000345(273 + T_a + 0.031 \times G_a)))).\\ k_3 : I_3 = 5.3 \times G_a(1 - exp((V_3 - 72(0.6 - \ldots\\ 0.0023(T_a + 0.031 \times G_a - 25)) + 0.45 \times I_3) \ldots\\ /72(0.000345(273 + T_a + 0.031 \times G_a)))).\end{array} \quad (6)$$

They do not give any information about the relation between the panels. The connections between panels have to be taken into account. The same current passes through $panel_1$, $panel_2$ and $panel_3$. Also, the total voltage V is the sum of V_1, V_2 and V_3. These connections are expressed by the constraints k_4, k_5, k_6 and k_7.

$$\begin{array}{l}(ok(connection1), k_4 : I_1 = I_2)\\ (-(connection_1), k_5 : I_2 = I_3)\\ (-(connection_1), k_6 : I_3 = I_{champ})\\ (ok(connection_2), k_7 : V_{champ} = \ldots\\ V_1 + V_2 + V_3)\end{array} \quad (7)$$

(7) represents the connection model for the system. This model has to be deduced from data provided by the installer. The sensors are modeled by the following constraints:

$$\begin{array}{l}(ok(sensor_1), k_8 : G_a = \tilde{G_a}\\ (ok(sensor_2), k_9 : T_a = \tilde{T_a}\\ (ok(sensor_3), k_{10} : I_1 = \tilde{I_1}\\ (ok(sensor_4), k_{11} : V_1 = \tilde{V_1}\\ (ok(sensor_5), k_{12} : V_2 = \tilde{V_2}\\ (ok(sensor_6), k_{13} : V_3 = \tilde{V_3}\\ (ok(sensor_7), k_{14} : I_{champ} = \tilde{I_{champ}}\\ (ok(sensor_8), k_{15} : V_{champ} = \tilde{V_{champ}}\end{array} \quad (8)$$

Requesting models from component database of the photovoltaic system, permits to get component models (5) and (6) together with the observation models (8). The photovoltaic system installer provides data that make it possible to establish (7).

The set of all the elementary models can be decomposed into several sets:

- E^{comp} that contains the elementary models with modes and without dataflows (for example, the constraint sets 5, 6 and 7).
- E^{obs} that contains the elementary models with modes and with dataflows, which contain terminal constraints (for example, the constraint set 8).

IV. Design of Testable Subsystems (TSSs) for Fault Diagnosis

The goal of a diagnosis system is to detect abnormal behaviors to avoid failures, breakdowns and damages and to propose hypotheses about the possible faulty components.

The faults in a physical system can be diagnosed by checking the consistency between knowledge represented by a behavior model and observations. In this research, the diagnosis system is a two-stage process: detection and isolation stages. In the first stage, the detection tests are computed from the model of the system to be diagnosed. This model is composed of a set of behavioral constraints, a set of terminal constraints containing dataflows. The principle used for testing is to form testable subsets of behavioral and terminal constraints. In the following, the testable subsets of constraints is called as testable subsystems (or TSSs). One can notice the equivalence between a detection test and a TSS.

In the second stage, a fault diagnosis analysis is performed using the symptoms generated during the previous stage.

Generally speaking, each detection test is related to a subset of behavioral constraints.

According to expressions (1), (2) and (3), a testable subsystem can be written as:

$$\begin{aligned} & \bigcup_{i\in T}\left(ok(c_i), k_i(obs, V_i) = 0\right) \\ \leftrightarrow \quad & \left(\bigwedge_{i\in T} ok(c_i), \bigcup_{i\in T} k_i(obs, V_i) = 0\right) \end{aligned} \tag{9}$$

By definition, if (9) is testable subsystems, it means that it exists a constraint k_T, depending only on data flows, such as $k_T(obs) = 0 \leftrightarrow \bigcup_{i\in T} k_i(obs, V_i) = 0$. Therefore, the test is given by:

$$\left(\bigwedge_{i\in T} ok(c_i), k_T(obs) = 0\right) \tag{10}$$

In the scientific literature, there are several methods for finding all the testable subsystems: the more general ones rely on a structuro-behavioral modeling. A method based on a bipartite graph approach has been used in ([3]), ([8]), ([25]). It makes it possible to compute the testable subsystems. Another method has been proposed in ([26]). This method is based on the retraction of sensors measuring the variables of a system. A method based on the Dulmage-Mendelsohn decomposition ([9]) has been proposed in ([15]). A general method for finding all the possible testable subsystems has been proposed in ([21]). This method is based on a structural analysis. It provides the constraints that have to be used for the design of each detection test and manages situations where constraints contain non deductible variables.

V. New method for symptom generation by using MILP problem et GLPK solver

The symptoms generation from the testable subsystems is a very important step for the fault diagnosis. In literature, there are several methods for symptom generation. The value propagation ([1]) and the constraints propagation ([24] are efficient methods for symptom generation. But, most approaches require the definition of propagation heuristics that depend on the problem to be solved whereas for fault diagnosis, general approaches are searched

In this section, a new method for symptom generation is presented. The main idea is to transform the constraints

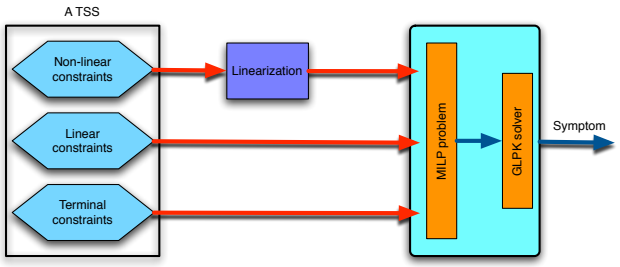

Fig. 3. Symptom generation by using the Milp problem and GLPK

of each TSS into a MILP Problem i.e a regular constraint satisfaction problem and then to use an MILP solver that check if each TSS has a solution or not. If the solver gives a solution then the test resulting from a TSS is positive and consequently, the behavior is consistent with reference model. If the solver does not find any solution then the test resulting from the TSS is negative and consequently, there is a inconsistency between observation and reference model.

MILP problems (Mixed Integer Linear Programming) make it possible to solve the linear or linearized problems. In order to formalize a MILP problem, a linear model describing the behavior of the system has to be formulated. This mathematical model has been implemented automatically in Java.

In order to solve the formalized problem, a MILP solver is needed. Several solvers are available:

- the GLPK solver (open-source software). This solver is suitable for medium problems.
- the CPLEX solver. This solver is suitable for great size problems

In this paper, the GLPK solver has been used. Figure 3 shows the necessary steps for symptom generation.

A. Linearization of a decreasing curve

The model of the photovoltaic system presented in figure 2 contains linear constraints (see relationships 7) and non-linear constraints (see relationships 5 and 6). Because the MILP can only manages linear or linearized problems, the constraints 5 and 6 have to be linearized (see figure 3). Because the curve of each panel is a decreasing curve, this section details the linearisation method of a decreasing curve (see figure 4).

Let consider a point (x, y) on the curve. The following constraints can be written:

$$\begin{aligned} x_1 \le x \le x_2 \leftrightarrow \quad & y = y_1 - \tfrac{y_1 - y_2}{x_2 - x_1}(x - x_1) \\ x_2 \le x \le x_3 \leftrightarrow \quad & y = y_2 - \tfrac{y_2 - y_3}{x_3 - x_2}(x - x_2) \end{aligned} \tag{11}$$

with $dom(x) = [x_1, x_3]$ and $dom(y) = [y_3, y_1]$.

By taking into account the variables δ_1 and δ_2, the constraints 11 can be rewritten by the following equations:

$$\begin{aligned} (\delta_1 = 1) \wedge (\delta_2 = 0) \leftrightarrow \quad & y = y_1 - \tfrac{y_1 - y_2}{x_2 - x_1}(x - x_1) \\ (\delta_1 = 1) \wedge (\delta_2 = 1) \leftrightarrow \quad & y = y_2 - \tfrac{y_2 - y_3}{x_3 - x_2}(x - x_2) \end{aligned} \tag{12}$$

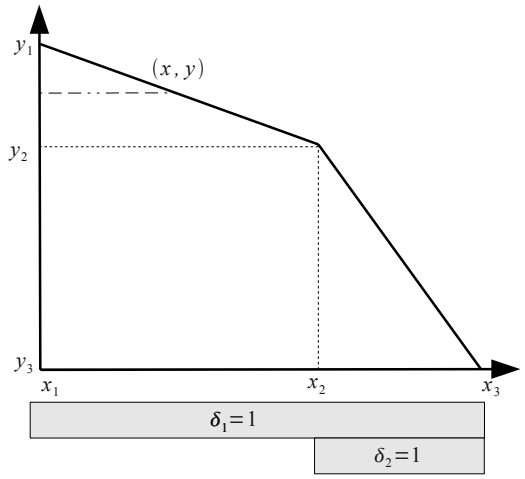

Fig. 4. A decreasing curve

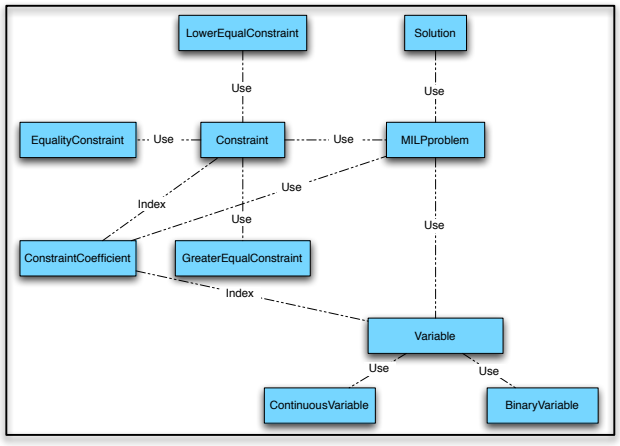

Fig. 5. Metamodel structure of the optimisation problem

or:

$$\begin{array}{ll} \delta_1 - \delta_2 = 1 \leftrightarrow & y = y_1 - \frac{y_1 - y_2}{x_2 - x_1}(x - x_1) \\ \delta_1 + \delta_2 = 2 \leftrightarrow & y = y_2 - \frac{y_2 - y_3}{x_3 - x_2}(x - x_2) \end{array} \quad (13)$$

Constraints (13) can be rewritten in a MILP format:

$$\begin{array}{ll} (y_1 - y) - \frac{y_1 - y_2}{x_2 - x_1}(x - x_1) \leq & (y_1 - y_3)(1 - \delta_1 + \delta_2) \\ (y_1 - y) - \frac{y_1 - y_2}{x_2 - x_1}(x - x_1) & \geq \frac{(y_2 - y_1)(x_3 - x_1)}{x_2 - x_1}(1 - \ldots \\ \delta_1 + \delta_2) & \\ (y_2 - y) - \frac{y_2 - y_3}{x_3 - x_2}(x - x_2) \leq & (y_1 - y_3)(2 - \delta_1 - \delta_2) \\ (y_2 - y) - \frac{y_2 - y_3}{x_3 - x_2}(x - x_2) \geq & \frac{(y_3 - y_2)(x_3 - x_2)}{x_3 - x_2}(2 - \ldots \\ \delta_1 - \delta_2) & \end{array} \quad (14)$$

In order to describe the variables δ_i, the following constraints are written:

$$\begin{array}{ll} \delta_1 = 1 \leftrightarrow & x_1 \leq x \\ \delta_2 = 1 \leftrightarrow & x_2 \leq x \end{array} \quad (15)$$

These constraints can be rewritten in a MILP format:

$$\begin{array}{l} x_1 - x \leq 0 \\ x_1 - x \geq (x_1 - x_3)\delta_1 \\ x_2 - x \leq (x_2 - x_1)(1 - \delta_2) \\ x_2 - x \geq (x_2 - x_3)\delta_2 \end{array} \quad (16)$$

Because all the values of δ_i are not possible, the following constraints have to be added:

$$\begin{array}{l} \delta_2 \leq \delta_1 \\ \delta_1 + \delta_2 \geq 1 \end{array} \quad (17)$$

Relationships (14), (16) and (17) represent the linearized equations of the decreasing curve.
This linearization approach has applied for equations (5) and (6).

B. MILP metamodel

A metamodel can be defined as a language from which a model can be generated. The figure 5 shows all the necessary classes for the model construction.

The computer implementation of the metamodel is done without user intervention, but obviously, the user should complete the metamodel with the necessary data. This is made easy by using external files of type xml.

C. GNU Linear Programming Kit (GLPK)

GLPK is free, open source software. It makes possible to solve mathematical programs. Specifically, it solves linear programs (**LP**) via revised simplex method and primal-dual interior point method. It also solves linear mixed-integer programs (MILP) via branch-and-bound algorithm, together with advanced cut routines.
GLPK is intended to solve large-scale problems. It provides an optional presolver, which transforms the problem into one that has better numerical properties for the simplex algorithm; this is particularly useful for large-scale problems. GLPK was developed, and is maintained, by Andrew Makhorin, Department for Applied Informatics, Moscow Aviation Institute.

VI. Diagnosis analysis

The diagnosis analysis aims at computing all the possible diagnoses from the symptoms provided by the detection tests. Different kinds of approaches may be used to analyze the symptoms. The decision tree approach is a general approach to analyze symptoms but not relevant for complex systems ([5]). Case-based reasoning compares the current symptoms with patterns coming from a knowledge database in order to retrieve a similar situation with the current one ([16]). This approach is not exhaustive and does not benefit of the knowledge about the behavioral modes of the system components. The signature based approach relies implicitly on a no exoneration assumption ([20]), which may lead to erroneous diagnoses. The bridge approach shows that the consistency-based reasoning can be used to analyze the symptoms coming from the detection tests depicted by the involved modes ([18]), ([20]). The bridge approach presented in ([20]) shows that a detection test based on a TSS, can be depicted as a set of modes of related components.

Depending on the way of testing, test (10) yields to:

$$\bigwedge_{i\in T} ok(c_i) \leftrightarrow \forall obs \in OBS, k_T(obs) = 0 \quad (18)$$

or

$$\bigwedge_{i\in T} ok(c_i) \rightarrow \forall obs \in \mathcal{O}, k_T(obs) = 0 \quad (19)$$

where $\mathcal{O} \subset OBS$.

Practical diagnosis relies on expressions like (19). For each, unsatisfied test $k_T(obs) \neq 0$ for a given $obs \in OBS$ yields:

$$k_T(obs) \neq 0 \rightarrow \bigvee_{i\in T} cfm(c_i) \quad (20)$$

where the set $\{cfm(c_i); i \in T\}$ is named explanation for T: $Expl(T)$.

If a test is satisfied, $k_T(obs) = 0$, it does not lead to any formal conclusion.

If a test has been performed for all the possible observations OBS, equation (20) becomes:

$$\bigwedge_{i\in T} ok(c_i) \leftrightarrow \forall obs \in OBS, k_T(obs) = 0 \quad (21)$$

But it is almost never possible to test all the possible situations OBS because:

- even if the variables are observable, the system has to be controllable to be able to check all the possible OBS
- sensors and actuators may be faulty, so it is not possible to prove that all the situations OBS have been checked.

Therefore, to compute diagnoses, expressions like (20) are used. A diagnosis is thus a conjunction of faulty modes such that it may explain all the current known test explanations. Then, in each diagnosis, the fault mode of every faulty component is taken into account. In other words, consider a set of inconsistent detection tests $\{T_1, \ldots, T_n\}$. Finding a diagnosis consists in searching a conjunction of modes which verifies the following expression:

$$\bigwedge_{T_i} \bigvee_{mode_j \in Expl(T_i)} mode_j \quad (22)$$

The second method which can be used to sort the diagnosis exploits the priori reliability of components: $p(mode(c_i) = ok(c_i))$. Then, because there are only two considered modes, $p(mode(c_i) = cfm(c_i)) = 1 - p(mode(c_i) = ok(c_i))$. Then, if $d = \bigwedge_i cfm(c_i)$ is a diagnosis, its a priori probability is given by:

$$\mu_T^a(d) = \prod_i (1 - p(mode(c_i) = ok(c_i)) \quad (23)$$

VII. Diagnosis strategy of photovoltaic system

In this section, the diagnosis strategy of photovoltaic systems is presented.

The list of the necessary sensors for diagnosis analysis are:

1) current sensor
2) voltage sensor
3) temperature sensor
4) pyranometer
5) laptop and diagnosis software

Suppose that the variables to be measured are determined, the following steps are necessary for fault detection and isolation:

1) the sensors measuring the variables G_a and T_a are installed in order to plot the curve of each photovoltaic panel in function of I and V. This step is very important to linearize the non-linear constraints modelling these panels.
2) the other sensors are consecutively installed (one by one, two by two....)
3) measurements of installed sensors are collected
4) these measurements are entered into the diagnosis software
5) the diagnosis software finds the testable subsystems
6) this software generates the symptoms by using the testable subsystems
7) this software finds all the possible diagnoses
8) if the goal is satisfied then the analysis diagnosis is finish, else, the steps 2, 3, 4, 5, 6 et 7 are repeated

Consequently, the diagnosis software provides all the possible diagnoses.

VIII. Application

In this section, the iterative diagnosis process presented in this paper is applied on the photovoltaic system (Fig. 2) in order to satisfy the following specifications: the faults on panels have to be isolated and the faults on connections have to be at least detected. The necessary sensors are already determined. The necessary sensors are already determined. The photovoltaic system with sensors is modelled by the constraint sets (5, 6, 7 and 8).

By applying the diagnosis strategy presented in section VII, all the sensors already determined have to be installed. Suppose that the measured values are: $T_a = 330$, $I_1 = 3.40$, $V_1 = 21.6851$, $V_2 = 28.9135$, $V_3 = 28.9135$, $V_{champ} = 79.5121$, $I_{champ} = 3.40$. Suppose also that the value of photocurrent related to the measured irradiation G_a is $I_{ph} = 3.77$.

By applying one of methods of testable subsystem design, 8 minimal testable subsystems (**TSS**s) are obtained. This **TSS**s are:

- $TSS_1 : k11, k3, k5, k8, k4, k15, k14$
- $TSS_2 : k12, k6, k5, k8, k4$
- $TSS_3 : k12, k11, k3, k6, k15, k14$
- $TSS_4 : k11, k10, k9, k7, k13$
- $TSS_5 : k11, k3, k10, k2, k5, k15, k14$
- $TSS_6 : k12, k6, k10, k2, k5, k15, k14$
- $TSS_7 : k9, k1, k8, k15, k14$
- $TSS_8 : k10, k2, k8, k4, k15, k14$

By applying the method of symptom generation, the following results are obtained: the tests resulting from the testable subsystems $SST1$, $SST2$, $SST3$, $SST4$, $SST5$, $SST6$, $SST8$ are positive (observations are consistent with models) and the test resulting from $SST7$ is negative (observations are inconsistent with models).

The symptoms found are used for diagnosis analysis. A

diagnosis software DXLAB (developed in the laboratory G-SCOP) has been used to find out all the possible diagnoses. These symptoms are declared in DXLab as in figure 6.

Once symptoms are declared, the diagnoses shown in figure

```
abed@ubuntu-pioneer:~$ dxlab
Welcome into DXLAB version 0.2.2
[1] test k3 k4 k5 k8 k11 k14 k15 positive
test t001 [enabled], symptom:0.0 % (positive)
  k3, k4, k5, k8, k11, k14, k15
[2] test k4 k5 k6 k8 k12 positive
test t002 [enabled], symptom:0.0 % (positive)
  k4, k5, k6, k8, k12
[3] test k3 k6 k11 k12 k14 k15 positive
test t003 [enabled], symptom:0.0 % (positive)
  k3, k6, k11, k12, k14, k15
[4] test k7 k9 k10 k11 k13 positive
test t004 [enabled], symptom:0.0 % (positive)
  k7, k9, k10, k11, k13
[5] test k2 k3 k5 k10 k11 k14 k15 positive
test t005 [enabled], symptom:0.0 % (positive)
  k2, k3, k5, k10, k11, k14, k15
[6] test k2 k5 k6 k10 k12 k15 k16 positive
test t006 [enabled], symptom:0.0 % (positive)
  k2, k5, k6, k10, k12, k15, k16
[7] test k1 k8 k9 k14 k15 negative
test t007 [enabled], symptom:100.0 % (negative)
  k1, k8, k9, k14, k15
[8] test k2 k4 k8 k10 k14 k15 positive
test t008 [enabled], symptom:0.0 % (positive)
  k2, k4, k8, k10, k14, k15
```

Fig. 6. The sysmptom declaration

7 were obtained.

Suppose that the sensors are reliable (faults on sensors are

```
[9] show diagnoses
#0 (score:100.0 %, apriori:10.0%, vote:100.0%)
  k1 is not ok

#1 (score:100.0 %, apriori:10.0%, vote:87.5%)
  k9 is not ok

#2 (score:100.0 %, apriori:10.0%, vote:62.5%)
  k8 is not ok

#3 (score:100.0 %, apriori:10.0%, vote:50.0%)
  k14 is not ok

#4 (score:100.0 %, apriori:10.0%, vote:37.5%)
  k15 is not ok
```

Fig. 7. The possible diagnoses

not considered), the diagnosis software shows that a fault has occured in the panel 1. Consequently, the problem has been isolated and the specifications are satisfied.

IX. Conclusions

This paper points out how iterative fault diagnosis system can be achieved. The necessary steps are detailed. The modelling aspect is firstly discussed. The information required to depict a system to be diagnosed has been presented. Then, all the possible testable subsystems are generated and the way of testing them has been presented. Testable subsystems are transformed into a MILP problem and a GLPK solver is used to illustrate a possible implementation. Finally, a photovoltaic system that makes it possible to clarify the iterative diagnosis process is presented.

References

[1] Krzysztof Apt. *Principles of Constraint Programming*. Cambridge University Press, 2003.

[2] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki. *Diagnosis and Fault-tolerant Control*. Springer-Verlag, 2006.

[3] M. Blanke, M. Kinnaert, and M. Staroswiecki. *Diagnosis and fault tolerant control*. Springer, 2003.

[4] J.P. Cassar, M. Staroswiecki, and V. Cocquempot. Optimal residual design for model-based fault detection and isolation. In *3rd European Control Conference ECC95*, Roma, Italy, 1995.

[5] M. Chen, Alice X. Zheng, Jim Lloyd, Michael I. Jordan, and Eric Brewer. Failure diagnosis using decision trees. volume 0, pages 36–43, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[6] L. Chittaro and R. Ranon. Hierarchical model-based diagnosis based on structural abstraction. *Artificial Intelligence*, 47(1-2):147–182, 2004.

[7] J. De Kleer and B. C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32:97–130, 1987.

[8] P. Declerck and M. Staroswiecki. Characterization of the canonical components of a structural graph for fault detection in large scale industrial plants. In *European Control Conference*, pages 298–303, Grenoble, France, 1991.

[9] A. L. Dulmage and N. S. Mendelsohn. A structure theory of bi-partite graphs of finite exterior extension. *Transactions of the Royal Society of Canada*, 53(III):1–13, 1959.

[10] P. Frank. Analytical and quantitative model-based fault diagnosis- a survey and some new results. *European Journal of Control*, 2:6–28, 1996.

[11] E. Frisk and M. Krysander. Sensor placement for maximum fault isolability. In *The 18th International Workshop on Principles of Diagnosis (DX-07)*, 2007.

[12] J. Gertler. Analytical redundancy methods in fault detection and isolation. In *International Conference on fault diagnosis Tooldiag'93*, 1993.

[13] W. Hamscher, L. Console, and J. De Kleer. *Readings in Model-Based Diagnosis*. Morgan Kaufmann, 1992.

[14] R. Isermann. Model-based fault detection and diagnosis-status and applications. In *16th symposium on Automatic Control in Aerospace (ACA'2004)*, 2004.

[15] M. Krysander, J. Åslund, and M. . Nyberg. An efficient algorithm for finding minimal overconstrained subsystems for model-based-diagnosis. *IEEE Transactions on systems, Man, and Cybernetics-Part A: Systems and Humans*, 38(1):197–206, 2008.

[16] D. McSherry. Interactive case-based reasoning in sequential diagnosis. *Applied Intelligence*, 14(1):65–76, 2001.

[17] M. Nyberg and M. Krysander. Combining ai, fdi, and statistical hypothesis-testing in a framework for diagnosis. In *IFAC Safeprocess'03*, Washington, U.S.A., 2003.

[18] M. Nyberg and Mattias Krysander. Combining AI, FDI, and statistical hypothesis-testing in a framework for diagnosis. 2003.

[19] R.J. Patton and J. Chen. A review of parity space approaches to fault diagnosis. In *IFAC SAFEPROCESS Symposium*, Baden-Baden, 1991.

[20] S. Ploix, S. Touaf, and J. M. Flaus. A logical framework for isolation in fault diagnosis. In *SAFEPROCESS'2003*, Washington D.C., U.S.A., 2003.

[21] S. Ploix, A.A. Yassine, and J.M. Flaus. An improved algorithm for the design of testable subsystems. In *The 17th IFAC WORLD CONGRESS*, Seoul, Korea, 2008.

[22] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[23] H. Ressencourt, L. Travé-Massuyès, and J. Thomas. Hierarchical modelling and diagnosis for embedded systems. In *SAFEPROCESS'2006*, aug. 30th-sep. 1st 2006.

[24] S. Russell and P. Norvig. *Artificial Intelligence, a modern approach, 2nd ed.* Prentice Hall, 2003.

[25] M. Staroswiecki and P. Declerck. Analytical redundancy in nonlinear interconnected systems by means of structural analysis. In *IFAC AIPAC'89 Symposium*, Nantes, France, 1989.

[26] L. Travé-Massuyès, T. Escobet, and X. Olive. Diagnosability analysis based on component supported analytical redundancy relations. *IEEE transactions on Systems, Man, And Cybernetics - Part A: Systems and Humans*, 36(6):1146–1160, November 2006.

[27] A. Yassine, S. Ploix, and J.-M. Flaus. A method for sensor placements taking into account diagnosability criteria. *Applied Mathematics and Computer Science*, 18(4), 2008.

[28] A. Yassine, S. Ploix, and J.-M. Flaus. Structural approach for sensor placement with diagnosability purpose. In *The 17th IFAC World Congress*, Seoul, Corea, 2008.